Robust Time-Optimal Path Tracking Control of Robots: Theory and Experiments

Aidan James Cahill

B.Sc. (Hons) Mathematics, Northumbria (UK)

December 1995

A thesis submitted for the degree of Doctor of Philosophy
of the Australian National University

Department of Systems Engineering
Research School of Information Sciences and Engineering
The Australian National University

Declaration

These doctoral studies were conducted with supervision from Dr. Matthew James, Dr. Jon Kieffer and Prof. Darrell Williamson, all of the Department of Engineering, Faculty of Engineering and Information Technology, The Australian National University.

The work contained in this thesis, except where explicitly stated, is original research performed by the author under the guidance of Matt, Jon and Darrell. This work has not been submitted for a degree at any other university or institution.

Much of the research contained in this thesis has been published in or submitted to conferences and journals as listed below.

Journal Papers:

[J1] A.J. Cahill, M.R. James, J.C. Kieffer and D. Williamson, "Remarks on the Application of Dynamic Programming to the Optimal Path Timing of Robot Manipulators", *Submitted to International Journal of Nonlinear and Robust Control.*

[J2] A.J. Cahill, J.C. Kieffer and M.R. James, "Robust Time-Optimal Path Tracking: Theory and Experiments", *Submitted to IEEE Transactions on Robotics and Automation.*

Conference Papers:

[C1] A.J. Cahill, M.R. James, J.C. Kieffer and D. Williamson, "Optimal Path Timing of Robot Manipulators via Dynamic Programming", *International Workshop on Nonlinear Systems and Adaptive Control*, Sydney, Australia, September 1994.

[C2] A.J. Cahill, J.C. Kieffer and M.R. James, "Fast Pick and Place at Robot Singularities", *1995 IEEE International Conference on Robotics and Automation*, Nagoya, Japan, May 1995.

[C3] A.J. Cahill, J.C. Kieffer and M.R. James, "Implementation Issues in the Time-Optimal Control of Robot Manipulators Along Specified Paths", *1995 National Conference of the Australian Robot Association*, Melbourne, Australia, July 1995.

[C4] A.J. Cahill, J.C. Kieffer and M.R. James, "On Representing Robot Modelling Errors as Disturbances in Joint Accelerations: Theory and Experiments", *34th IEEE International Conference on Decision and Control*, New Orleans, USA, December 1995.

[C5] A.J. Cahill, J.C. Kieffer and M.R. James, "Time-Optimal Path Tracking to a Specified Tolerance in the Presence of Plant Uncertainty", *1996 IASTED International Conference on Applications of Control and Robotics*, Orlando, USA, January 1996.

[C6] A.J. Cahill, J.C. Kieffer and M.R. James, "Robust Time-Optimal Trajectory Planning for Robot Manipulators", *Submitted to 1996 IEEE International Conference on Robotics and Automation*, Minneapolis, USA, June 1996.

[C7] A.J. Cahill, J.C. Kieffer and M.R. James, "Closed-Loop Trajectory Generation for Robust Time-Optimal Path Tracking", *Submitted to 1996 IEEE International Conference on Robotics and Automation*, Minneapolis, USA, June 1996.

Canberra, December 1995.

A. J. Cahill.

Aidan James Cahill,
Department of Systems Engineering,
Research School of Information Sciences and Engineering,
The Australian National University,
Canberra ACT 0200, AUSTRALIA.

Acknowledgements

I would like to thank my supervisors, Matt James and Jon Kieffer, for all of their efforts, encouragement and support over the last three and a half years. That I could have had them as supervisors and as friends has been an enormous blessing. Thanks guys. I would also like to thank John Moore and Darrell Williamson who have supported and encouraged me greatly. And I would like to thank Iven Mareels and Sylvia for the friendship and help that they have given me throughout my time in Australia. ANU is truly blessed to have such "people" people inside its doors.

I would like to thank Dave Bennett, Stan Scott, Ron Atkinson and Mike Burke who inspired me as an undergraduate, who taught me that research could be fun, and who encouraged and supported my decision to do a PhD.

To all the people in the Departments of Systems Engineering and Engineering, thanks for your friendships. They have been very much appreciated. A very special thanks goes to my very special Christian brothers - Degsy, Jezza, 'nuth and Perry - who have endured much, yet who have supported and encouraged me far more.

My thanks also to the many friends outside of the ANU in whom meeting my life has been made richer. I thank Julie, Dave and the Fusion crew who let me share their visions. I also thank those at St Christopher's, at "Uni" Church and in Focus who have shared innumerable words of wisdom and encouragement. And thanks to Jon and Lyn for the south-east coast, Phil Collins and my Christmas lunches!

Finally, to my dear Mum and Dad, who have worked all their lives and forsaken much to give my brothers and I every opportunity in life, thank you. I love you both very very much.

"You are worthy, our Lord and God,

to receive all glory and honour and power,

for you created all things,

and by your will they were created and have their being."

Revelation 4:11

All I have comes from you, and to you I give all.

Abstract

In this thesis, we address three problems concerned with the time-optimal control of robot manipulators; trajectory planning using dynamic programming, robust time-optimal path tracking control and fast pick and place at robot singularities.

We apply dynamic programming to solve the optimal path timing problem in robotics. Our approach differs to previous dynamic programming approaches to this problem in that we solve numerically the continuous optimal-control problem specified by the *Hamilton-Jacobi-Bellman* partial differential equation of dynamic programming using finite difference Markov chain approximations. We evaluate the applicability of such solutions using a real SCARA manipulator, and we analyse issues relating to the convergence of the dynamic programming approach.

We provide a systematic way of controlling an industrial robot to achieve accurate and time-optimal tracking of specified paths, in spite of unmodelled dynamics. We develop a theory which relates modelling errors, identified experimentally as disturbances in joint accelerations, to the performance of robots under computed-torque control. This theory is used to predict the levels of torque that need to be held on reserve during trajectory planning and the controller gains required to reject the disturbances to a specified tolerance. Experiments show that if these levels of torque are held in reserve and if the controllers are tuned in this way, then the robot performs near time-optimal tracking of the path, accurate to the specified tracking tolerance. We extend this method to a less conservative closed-loop architecture for on-line trajectory generation. We propose a feedback law for setting the reference path acceleration that is nearly time-optimal, robustly controllable, and accurate to a prescribed tolerance. Experiments confirm that the tracking times are reduced compared to the first approach, while the tracking accuracy and robustness remain approximately the same.

Finally, we demonstrate that singular configurations may be good sites with respect to reducing cycle times for pick and place operations, and develop a method which guarantees that the end-effector remains in the neighbourhood of the pick and place site long enough to perform the pick or place task.

Contents

Chapter 1

Introduction

1.1 Background

The time-optimal control of robot manipulators has been a major area of research for many years. This research is motivated by economic factors, primarily the desire of industry to maximise its profits through increased productivity and reduced costs. A major component of production costs is due to the cycle time of manufacturing operations. The cost of each operation is proportional to the time the robot spends doing it. Thus, because the added value remains the same, profits are increased by reducing cycle times.

There are generally two different types of motion control problems: *point-to-point* and *path-constrained.* The first type requires only that the end-effector pass through some initial and final positions (and velocities) and includes applications such as pick and place operations. The second type of problem requires that the end-effector motion be constrained to a prescribed path. This category includes applications such as laser or high-pressure water jet cutting, welding, gluing and spraying. In addition, there are many problems between these two extremes, such as point-to-point motion control in an environment filled with obstacles.

A robot's timed motion is referred to as its "trajectory". The *trajectory* which results in point-to-point motion control is traditionally unplanned. In most other cases, the robot's trajectory is planned.

A variety of algorithms have been developed for planning and controlling the motion of robots. When the motion is constrained to a path, it has been usual to sub-divide the control problem into three separate levels; namely *path planning*, *trajectory planning* and *trajectory tracking*, Figure 1.1. The reason for dividing the problem in this way is that the collective problem is very complicated to solve because robot dynamics are usually highly nonlinear and coupled, and the control problem is of high dimension with multiple inputs and outputs.

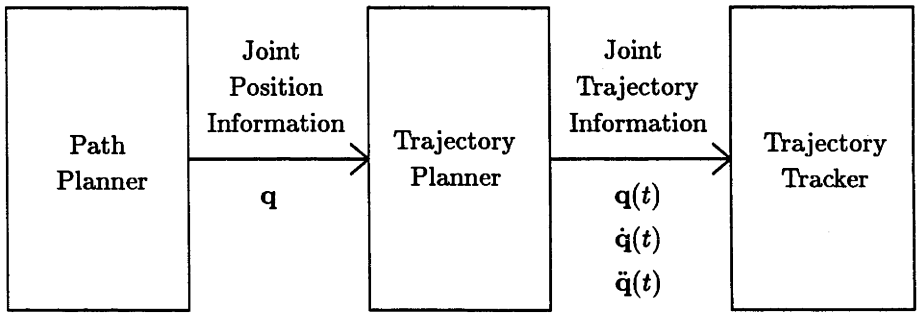


Figure 1.1: The 3 Stages of Robot Motion Control

Path planning is concerned with the geometry of the desired task. The path planner would take into account kinematic constraints on the task, for example to ensure that obstacles are avoided, but might also consider dynamic constraints, for example to avoid configurations at which excessively high forces might be generated. The output of the path planner will be the joint position information, $\mathbf{q}$, which may be represented either as a discrete set of points or via one or more continously parameterised functions. Many different path planning schemes have been proposed. However, this area is not the focus of our research and we would refer the reader interested in such schemes to, for example, [49] and references therein.

The trajectory planner associates time with the joint path information to yield a time history, or trajectory of the desired joint positions, velocities and perhaps accelerations. In principle, the resulting timed joint information might be specified continuously, but most methods provide this information discretely since the trajectory tracker

operates in discrete time.

There have been two main approaches to minimum-time trajectory planning, reflecting the two main approaches to optimal control generally. The first is based on the Pontryagin Maximum Principle [52] (such methods are sometimes referred to as *shooting methods* in the path timing literature) and the second involves dynamic programming. A more detailed discussion of these approaches is given in the descriptions of the problems that we consider which follows in § 1.2. A comprehensive survey summarising many of the results in the trajectory planning area can be found in [59].

In principle, the path and/or trajectory planning might be performed on-line. However, for anything more than the simplest of tasks, the computational complexity involved presently requires that this planning be done off-line.

The trajectory tracker is a part of the robot system. It consists of two parts; the trajectory generator and the tracking controller. The trajectory generator provides to the tracking controller the reference data calculated by the trajectory planner. The purpose of the tracking controller is to make the robot's position and velocity match the reference position and velocity. Many different tracking controllers have been proposed, see for example [1] and references therein, although most often in practice a simple linear (PID) controller is used. Whilst the nonlinearities of the manipulator dynamics are not taken into account, such trackers can generally keep the manipulator reasonably close to the desired trajectory.

We note that in contrast to earlier approaches for point-to-point motion control, some more recent work has combined the path and trajectory planning problems of this divided approach in order to obtain time-optimal trajectories for point-to-point motion control [25, 26, 32, 39, 56, 57]. These schemes have developed efficient search techniques in order to overcome the complexities involved in this unified approach.

In this thesis, we have considered three problems concerned with the time-optimal motion control of robots:

- *The application of dynamic programming to the problem of planning time-optimal trajectories for a robot manipulator whose motion is constrained to a path.*
- *The time-optimal control of robot manipulators along specified paths and to a specified tolerance in the presence of plant uncertainty.*
- *To investigate whether singular configurations would be good sites with respect to reducing the cycle times of pick and place operations.*

These problems are now discussed in more detail.

1.2 The Problems

Problem 1: Trajectory Planning using Dynamic Programming

To use dynamic programming to plan the time-optimal motions of robot manipulators constrained to specified paths.

Several approaches to the minimum-time trajectory planning problem noted that the path constraint allows a reduction in the dimension of the state of the problem. The reduction is from typically 12, the joint positions and their derivatives, to 2, the path displacement and velocity [8, 51, 58, 60, 66]. This made calculation of a theoretically time-optimal trajectory feasible. Prior to this, assumptions and simplifications had to be made in order to solve the problem for "near" time-optimal solutions, [36, 42, 43, 71].

Minimum-time solutions were first proposed (independently) by Bobrow *et al.* [8] and Shin and McKay [60] in 1985. Their methods were based on Pontryagin's Maximum Principle which shows that the control achieving the minimum-time solution will be *bang-bang* in nature, i.e. will take either its maximum or minimum value at all times [52]. They are sometimes referred to as *shooting methods* because solution is achieved by "shooting" trajectories in the path position-path velocity phase-plane. Subsequent literature has appeared which is directed towards making the schemes more computationally efficient [66], and dealing with *discontinuity points* [66], *critical points* [51], and

with *singular points* [54, 58].

Noting the dimension reduction, other authors recognised the potential for using the dynamic programming method to solve the minimum-time trajectory planning problem [51, 61, 64]. The real advantage in using dynamic programming is not that it gives better solutions, but rather that it readily allows solutions for criteria other than pure minimum-time. While Pontryagin's Maximum Principle can also handle such criteria, solution of the resulting multi-dimension 2-point boundary value problem is often less than trivial, see for example [55] where Shiller considers a "time-energy" performance criteria. The approaches in [51, 61, 64] solve the reduced-dimension problem using Bellman's recursion equation [7]. This equation is discrete in the states and in time, and involves the integration of the system dynamics over a fixed time interval.

In our approach, we also consider the application of dynamic programming to the problem of the planning of the time-optimal motion of robot manipulators along specified paths. However, we approach the problem from a different aspect than other dynamic programming approaches to this problem. We consider the numerical solution of the continuous optimal-control problem, specified by the Hamilton-Jacobi-Bellman partial differential equation [41] using finite difference Markov chain type approximations. Our desire is to evaluate the applicability of the resulting solutions to a real manipulator and to analyse issues relating to the convergence of the approach.

Problem 2: Robust Time-Optimal Path Tracking Control

To provide a systematic way of ensuring the accurate and time-optimal tracking of the specified path by a robot manipulator.

Solutions resulting from traditional approaches to the time-optimal trajectory planning problem are theoretically time-optimal but if applied to a real robot, the robot may fail to track them because they may be too demanding [8, 51, 54, 58, 60, 61, 64, 66]. The main reasons for this are that the design methods assume exact knowledge of the

dynamics and that they take no account of the dynamics of the tracking controller which is used to reject the tracking errors.

Some more recent literature has attempted to deal with these issues. The first approach involves quantifying the model parameter uncertainties and then making allowances for them when planning the trajectory. Shin and McKay [62] devised an algorithm for planning trajectories that are robust to given payload uncertainties. They convert bounds on the payload uncertainty to bounds on the torque uncertainty and then plan time-optimal trajectories robust to the payload uncertainty by holding this amount of torque in reserve. Huang and Chen [34] also devise a scheme for dealing with payload uncertainty. This scheme has two parts. Off-line they calculate and tabulate the switching times for the problem based on several payload masses. On-line, and based on the system response, the payload is then estimated and the appropriate switching times chosen. Given its adaptability, the scheme in [34] has the ability to be less conservative than that in [62].

Another approach which can cope with small parameter uncertainty was proposed by Tam [68]. This is based upon a perturbation scheme which is applied in feedback and which amends the switching times to allow authority to control.

More recently, there has been a push towards considering robust control techniques, including $\mathcal{H}_\infty$ approaches, for the robust control of manipulators, [4, 19, 44, 53, 70] for example. However, very little of this work has focussed on the time-optimal control problem, which is more difficult to solve because the actuators are operating at their limits. The only contribution to date has been a theoretical approach by Lyashevskiy and Chen [50] who use Bellman-Lyapunov theory to find a closed-form solution for the optimal control in a point-to-point minimum-time control problem. It is interesting that their approach uses a non-minimum-time performance index.

The second approach involves modifying the trajectories on-line in order to avoid the loss of control due to the model parameter uncertainties [2, 3, 22, 23, 24]. The approach by Dahl and Nielsen [22, 23, 24] implements a *path velocity controller* on-line to *time-*

scale (slow down or speed up) the nominal path trajectory. The nominal path trajectory is calculated using one of the shooting or dynamic programming methods described above. The approach consists of two parts. The desired *on-line* path acceleration is calculated as a function of the nominal path velocity and acceleration, and of the on-line path position and velocity. Bounds are then calculated on the on-line path acceleration. These are based on the on-line path position and velocity, and on measured data. Next, the on-line trajectory information is updated by integrating the path acceleration, saturated by the on-line bounds if necessary. The on-line bounds on the acceleration are designed so that the commanded torques will not be clipped, so that the end effector remains on the desired path. However, because the bounds are calculated from measured data, there can be no guarantee that any valid bounds will exist, in which case the commanded torques will be clipped and a loss of tracking will occur. Further, the desired path acceleration might be such that no actuator is fully utilised. To overcome this, they introduce the idea of *time-scaling* of the nominal trajectory. This idea provides a time shifting of the nominal trajectory so that it is less or more demanding as required. Together, these ideas seem to provide good results, although no consideration is made of the accuracy of tracking and the method requires the tuning of filter parameters in an apparently ad hoc fashion.

Dahl and Nielsen's approach was later modified by Arai *et al.* [2, 3] to resolve the tracking error into components tangential and orthogonal to the direction of motion, and to place a higher priority on reducing the orthogonal component. Further, they implemented an observer to reduce the amount of computation of the nonlinear dynamics required by the method.

Whilst all of this research is concerned with the problem of path tracking, none considers how to ensure tracking to a specified tolerance. While "good" tracking might be an acceptable criteria for a paint spraying task, for example, it might not be sufficient for, say, a cutting operation, which might require accuracy of tracking down to single millimetres. This provides the main motivation for our research; to provide a systematic way of ensuring the accurate and time-optimal tracking of a specified path.

Problem 3: Fast Pick and Place at Kinematic Singularities

To investigate whether singular configurations would be good sites with respect to reducing the cycle times of pick and place operations.

This piece of work is also concerned with fast robot motion, but considers a point-to-point motion control problem.

We hypothesise that singular configurations might be good sites at which to perform pick and place operations, because they offer the potential for reducing cycle times versus using neighbouring regular configuration sites. This is based on the observation that the end effector can be brought to rest at a singular configuration without stopping the mechanism, allowing the robot to keep some of its kinetic energy while performing the pick or place task.

Our aim is to investigate this hypothesis.

We believe that this idea is original and so we have no literature with which to compare our ideas.

1.3 Summary of Contributions

Trajectory Planning using Dynamic Programming

In this work we considered applying dynamic programming to solve the problem of the optimal path timing of robot manipulators. Our approach differed to previous dynamic programming approaches to this problem in that we solved numerically the continuous optimal-control problem specified by the *Hamilton-Jacobi-Bellman* partial differential equation of dynamic programming using finite difference Markov chain type approximations. Our desire was to evaluate the applicability of the resulting solutions to a real manipulator and to analyse issues relating to the convergence of this approach.

We conclude that it is feasible to use dynamic programming to solve the optimal path timing problem in robotics. Further, we note that:

- We have applied solutions of the dynamic programming algorithm to a real SCARA manipulator with good results
- We have shown that the rate of convergence of the numerical scheme is consistent with that predicted by theory.
- We have found that the minimum-time end-point (target) constraint imposes severe limitation regarding the rate of convergence and computational speed of the dynamic programming algorithm (in particular, commonly used acceleration methods do not work).
- If a pure minimum-time criteria is required, the Pontryagin Maximum Principle based shooting method is superior and is the method of choice. However, the dynamic programming algorithm can easily handle more general optimisation criteria, in which case the advantages of the PMP approach diminish.

Robust Time-Optimal Path Tracking Control

In this work our goal was to provide a systematic way of controlling industrial robots to achieve accurate and time-optimal tracking of specified paths. The major issue that we addressed was to take previously developed theory and to develop ways of making it practical, to apply to real industrial robots. In particular, our focus has been to address the issue of how a user specified tracking tolerance can be achieved in spite of unmodelled dynamics.

To this end, we have developed three major results:

- Based on representing modelling errors as disturbances in joint accelerations, we have developed a theory for relating these disturbances to the performance of a robot under computed-torque control. Experimental results confirmed that the theory works well in practice.
- We have used this theory in a predictive way to provide a method for planning trajectories which are robust to modelling disturbances. The method identifies

the disturbance using a trajectory close to the time-optimal trajectory being sought. The theory is then used to predict the torques that need to be held on reserve during trajectory planning and the controller gains required to reject the disturbances to a specified tolerance. Experiments have shown that if trajectories are planned and the controllers tuned in this way, then the robot performs near time-optimal tracking of the path, accurate to the specified tracking tolerance.

- We have extended this method to a less conservative closed-loop architecture which generates the trajectory on-line. We have proposed a feedback law for setting the reference path acceleration such that path tracking is nearly time-optimal, robustly controllable, and accurate to a prescribed tolerance. Experimental results have demonstrated that the tracking times are reduced compared to the first approach, while the tracking accuracy and robustness remain approximately the same.

Fast Pick and Place at Kinematic Singularities

In this work we investigated whether singular configurations might be good sites, with respect to reducing cycle times, for pick and place operations.

- We have shown that cycle times of pick and place operations can be reduced by placing the pick and place sites at kinematic singularities, compared to nearby regular configurations.

- We have developed a method which guarantees that the end-effector remains within a given neighbourhood of the pick and place site for a given amount of time - long enough to perform the pick or place task.

Chapter 2

Remarks on the Application of Dynamic Programming to the Optimal Path Timing of Robot Manipulators

Abstract

In this chapter, we investigate the use of the dynamic programming approach in the solution of the optimal path timing problem in robotics. This problem is computationally feasible because the path constraint reduces the dimension of the state in the problem to 2. The Hamilton-Jacobi-Bellman equation of dynamic programming, a nonlinear first order partial differential equation, is presented and is solved approximately using finite difference methods. Numerical solution of this results in the optimal policy which can then be used to define the optimal path timing by numerical integration. Issues relating to the convergence of the numerical schemes are discussed, and the results are applied to an experimental SCARA manipulator.

2.1 Introduction

Optimisation methods are commonly used in engineering design. In the context of systems theory, two main approaches to optimal control have emerged: (i) Pontryagin

Maximum Principle (PMP) [52], (ii) Dynamic Programming (DP) [7]. The PMP is a set of necessary conditions that an optimal control (if one exists) must satisfy, and is used to identify candidate open loop optimal controls. Dynamic programming involves the use of a value function and provides a means for determining optimal state feedback controls (verification theorem).

In many problems of interest in nonlinear control a state feedback controller is desired, and often such problems lend themselves to potential solution via dynamic programming, such as minimax (including $\mathcal{H}_\infty$) control and stochastic optimal control. The usual difficulty with dynamic programming is the well known "curse of dimensionality" [7], and this limits the utility of DP in practice. Even if the optimal solution cannot be found (due to prohibitive computational costs), knowledge of the theoretically optimal solution can serve as a useful guide in designing implementable suboptimal controllers. However, there are some applications where the DP method is feasible. Such applications are of interest because they afford an opportunity to study practical issues relating to the DP method. Such issues include approximate (i.e. numerical) computation of the optimal state feedback controls, properties of the algorithms such as rate of convergence and memory requirements, and also real-time implementation issues. The purpose of this work is to study some of these issues in the context of optimal path timing for robot manipulators.

The optimal path timing problem has been studied extensively in the past. Typically, shooting techniques based on the PMP are employed, for example Bobrow *et al.* [8] and Shin and McKay [60]. These methods assume knowledge of the bang-bang nature of the optimal controls. Thus the DP and PMP methods can be directly compared in this application. Other papers using DP for this application include Pfieffer and Johanni [51] and Shin and McKay [61], although they solve the method in a different manner to that presented below.

The contents of the chapter are as follows. In §2.2 we define the optimal control problem, and in §2.3 we construct a second order state space system (the reduced

system) with one control input, state dependent constraints on this input and state constraints, and reformulate the optimal control problem. In §2.4 we use dynamic programming methods to solve the optimal control problem, and present a numerical approximation to the solution based on the numerical solution of the Hamilton-Jacobi-Bellman (HJB) partial differential equation. This numerical solution employs discretisation of the state space and so provides discrete approximations to the continuous solution. Solution of the HJB equation provides the value function (the *minimum-time function* in the pure minimum-time case), the associated optimal feedback control policy (a feedback function of the path position and velocity), and information concerning the controllability of the system. In §2.5 we present two examples based on a SCARA manipulator that we have in our laboratory. The first is a pure minimum-time example, and the second an example using a minimum-time plus state and control energy criteria which serves to demonstrate the versatility of the dynamic programming method when dealing with problems which have non bang-bang solutions. In both examples, we implement the resulting optimal solutions on the SCARA manipulator to assess the applicability of the method. In §2.6 we discuss the computational issues involved in the calculation and implementation of the solutions, and in §2.7 we consider the effects of adding a discontinuous friction model into the system dynamics.

2.2 The Time-Optimal Path Tracking Problem

Consider a generalised n degree of freedom, rigid-body non-redundant robot. The dynamic equations may be written as

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{n}(\mathbf{q}, \dot{\mathbf{q}}) = \boldsymbol{\tau} \tag{2.1}$$

where $\mathbf{q} \in \mathbb{R}^n$ are the joint variables, $\boldsymbol{\tau} \in \mathbb{R}^n$ are the joint actuator forces or torques whose constraints, $\underline{\boldsymbol{\tau}}(\mathbf{q}, \dot{\mathbf{q}}) \leq \boldsymbol{\tau} \leq \overline{\boldsymbol{\tau}}(\mathbf{q}, \dot{\mathbf{q}})$, may be position and/or velocity dependent, $\mathbf{M}(\mathbf{q}) \in \mathbb{R}^{n \times n}$ is the inertia matrix, and $\mathbf{n}(\mathbf{q}, \dot{\mathbf{q}}) \in \mathbb{R}^n$ is the vector containing the friction, coriolis, centrifugal force and gravity terms.

The position of the end-effector in the task space can be described in terms of the

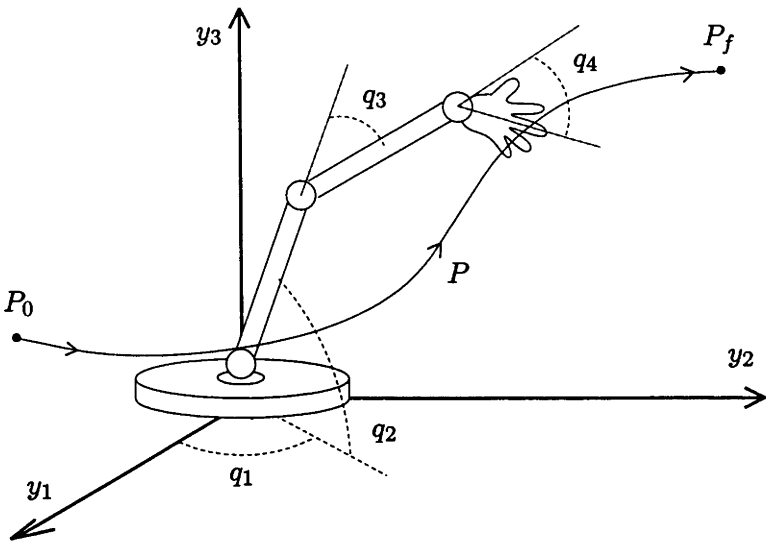


Figure 2.1: The Path To Be Traversed

joint variables through the kinematic mapping $\mathbf{y} = \mathbf{r}(\mathbf{q})$, and the time-optimal problem is to find the joint torques $\boldsymbol{\tau}(t)$, $0 \leq t \leq t_f$, so that the end-effector traverses a task space path P as fast as possible.

Assume that P is given as a regular, parametric curve, $\mathbf{y} = \mathbf{p}(s)$, $\mathbf{p} \in C^2$, parameterised by the scalar s, $s_0 \leq s \leq s_f$. Then, $\mathbf{q}$ and s are related by the forward kinematic equation

$$\mathbf{r}(\mathbf{q}) = \mathbf{p}(s) \tag{2.2}$$

which we assume can be solved through inverse kinematics for $\mathbf{q} = \mathbf{f}(s)$.

Assuming an explicit and differentiable expression for $\mathbf{f}(s)$ can be determined, then the system dynamics (2.1) can be re-expressed in terms of s rather than $\mathbf{q}$ as

$$\mathbf{a}(s)\ddot{s} + \mathbf{b}(s, \dot{s}) = \boldsymbol{\tau} \tag{2.3}$$

with $\underline{\boldsymbol{\tau}}(s, \dot{s}) \leq \boldsymbol{\tau} \leq \overline{\boldsymbol{\tau}}(s, \dot{s})$, [8, 51, 58, 60]. This form describes how the control inputs $\boldsymbol{\tau}$ are related to the path displacement, s, velocity, $\dot{s}$, and acceleration, $\ddot{s}$, and it should be noted that all n equations of (2.3) must be satisfied simultaneously for the end-effector to track the specified path. (It is usually difficult to derive $\mathbf{f}(s)$ as an explicit differentiable function and derivatives are normally evaluated based on differentiation

of equation (2.2), Slotine and Yang [66].)

The optimal control problem is now defined as:

Find the control $\tau(t)$, $\underline{\tau}(s(t),\dot{s}(t)) \leq \tau(t) \leq \overline{\tau}(s(t),\dot{s}(t))$, *which minimises the performance measure*

$$J(s(0),\dot{s}(0)) = \int_0^{t_f} L(s(t),\dot{s}(t),\ddot{s}(t))\, dt\,, \tag{2.4}$$

$L(s,\dot{s},\ddot{s}) \geq 0$ *and* $L(s,\dot{s},\ddot{s}) \in C^2$, *subject to the dynamics* (2.3) *and to the boundary conditions* $(s(0),\dot{s}(0)) = (s_0,\dot{s}_0)$ *and* $(s(t_f),\dot{s}(t_f)) = (s_f,\dot{s}_f)$.

Note: In the pure minimum-time case, $L(s(t),\dot{s}(t),\tau(t)) = 1$, which yields $J(s(0),\dot{s}(0)) = t_f$ in (2.4).

2.3 Reformulation of the Problem

In this section, we reformulate the problem into an equivalent second order state space form with a single control input, state dependent constraints on the input, and state constraints, [8, 51, 60, 64, 66].

Let $x_1 = s$ and $x_2 = \dot{s}$. Then $\dot{x}_1 = x_2$ and $\dot{x}_2 = \ddot{s}$, and equations (2.3) may be rewritten in the equivalent nonlinear state space form:

$$\begin{bmatrix} 1 & 0 \\ 0 & a_1(x_1) \\ 0 & a_2(x_1) \\ . & . \\ 0 & a_n(x_1) \end{bmatrix} \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} x_2 \\ -b_1(\mathbf{x}) \\ -b_2(\mathbf{x}) \\ . \\ -b_n(\mathbf{x}) \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & \cdot & 0 \\ 1 & 0 & 0 & \cdot & 0 \\ 0 & 1 & 0 & \cdot & 0 \\ . & . & . & . & . \\ 0 & 0 & 0 & \cdot & 1 \end{bmatrix} \begin{bmatrix} \tau_1 \\ \tau_2 \\ . \\ \tau_n \end{bmatrix} \tag{2.5}$$

or

$$\mathbf{E}(\mathbf{x})\dot{\mathbf{x}} = \mathbf{A}(\mathbf{x}) + \mathbf{B}\tau. \tag{2.6}$$

Remark: Equation (2.6) differs from the usual nonlinear state space representation due to presence of the $\mathbf{E}(\mathbf{x})$ matrix premultiplying $\dot{\mathbf{x}}$. This representation is known as a nonlinear *descriptor system* form.

From the structure of (2.5), we observe that we might row reduce equations (2.5)

into a system equivalent form such that upper 2×2 sub-matrix of $\mathbf{E}(\mathbf{x})$ is diagonal and the rest is zero, but are prevented from using any one of the latter n rows as the pivotal row because $a_i(x_1)$ may equal zero for some x_1. To circumvent this problem, we define the path acceleration, $\ddot{s}$, to be a function of a new and independent control τ_0,

$$\ddot{s} \;=\; \dot{x}_2 \;\stackrel{\triangle}{=}\; \tau_0\,, \tag{2.7}$$

augment equations (2.5) with (2.7), and row reduce the latter n equations of the augmented system using the augmenting equation as the pivotal equation which allows the system (2.3) to be represented as follows:

1. Second Order System:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tau_0$$

2. State and Control Constraints:

$$\begin{bmatrix} b_1(\mathbf{x}) \\ \cdot \\ b_n(\mathbf{x}) \end{bmatrix} + \tau_0 \begin{bmatrix} a_1(x_1) \\ \cdot \\ a_n(x_1) \end{bmatrix} = \begin{bmatrix} \tau_1 \\ \cdot \\ \tau_n \end{bmatrix} \tag{2.8}$$

3. Control Constraints:

$$\underline{\tau_i}(\mathbf{x}) \le \tau_i \le \overline{\tau_i}(\mathbf{x}), \quad i = 1 \text{ to } n.$$

Controls τ_1,...,τ_n can be eliminated from the problem by considering that equations (2.8) can be satisfied for $\mathbf{x}$ if and only if

$$\underline{\tau_i}(\mathbf{x}) \le b_i(\mathbf{x}) + a_i(x_1)\tau_0 \le \overline{\tau_i}(\mathbf{x}), \quad i = 1 \text{ to } n. \tag{2.9}$$

In (2.9), there are two possibilities, depending on whether $a_i(x_1) = 0$ or $a_i(x_1) \neq 0$. Let $I(x_1) = \{\, i \in \{1, 2, ..., n\} : a_i(x_1) \neq 0 \,\}$.

If $a_i(x_1) = 0$, inequalities (2.9) can be written as state constraints $\underline{\tau_i}(\mathbf{x}) \le b_i(\mathbf{x}) \le \overline{\tau_i}(\mathbf{x})$. For a given x_1, the set of all x_2 simultaneously satisfying all such inequalities, denoted $\mathcal{A}_z(x_1)$, is

$$\mathcal{A}_z(x_1) = \bigcap_{i \notin I(x_1)} \{x_2 : \underline{\tau_i}(\mathbf{x}) \le b_i(\mathbf{x}) \le \overline{\tau_i}(\mathbf{x})\} \quad ,$$

and the region all $\mathbf{x}$ simultaneously satisfying all such inequalities, denoted $\mathcal{A}_z$, is

$$\mathcal{A}_z = \bigcup_{x_1} \mathcal{A}_z(x_1) \, .$$

Note that the inequalities defining $\mathcal{A}_z$ place no restrictions on τ_0. However, if $\mathbf{x} \notin \mathcal{A}_z$ then the motion of the end-effector will not be constrained to the path and there will exist no τ_0 which will avoid this.

On the other hand, if $a_i(x_1) \neq 0$, the remaining inequalities of (2.9) can be rearranged to give

$$-\frac{b_i(\mathbf{x})}{a_i(x_1)} + \frac{\underline{\tau_i}(\mathbf{x})}{|a_i(x_1)|} \leq \tau_0 \leq -\frac{b_i(\mathbf{x})}{a_i(x_1)} + \frac{\overline{\tau_i}(\mathbf{x})}{|a_i(x_1)|} \, . \tag{2.10}$$

In order that the end-effector tracks the path, there must exist at least one τ_0 satisfying all inequalities (2.10) simultaneously. Thus, the inequalities (2.10) act as state constraints. For a given x_1, the set of x_2 simultaneously satisfying inequalities (2.10), denoted $\mathcal{A}_{nz}(x_1)$, is

$$\mathcal{A}_{nz}(x_1) = \left\{ x_2 : \max_{i \in I(x_1)} \left\{ -\frac{b_i(\mathbf{x})}{a_i(x_1)} + \frac{\underline{\tau_i}(\mathbf{x})}{|a_i(x_1)|} \right\} \leq \min_{i \in I(x_1)} \left\{ -\frac{b_i(\mathbf{x})}{a_i(x_1)} + \frac{\overline{\tau_i}(\mathbf{x})}{|a_i(x_1)|} \right\} \right\} .$$

The region of all $\mathbf{x}$ simultaneously satisfying inequalities (2.10), denoted $\mathcal{A}_{nz}$, is

$$\mathcal{A}_{nz} = \bigcup_{x_1} \mathcal{A}_{nz}(x_1) \, .$$

In turn, for $\mathbf{x} \in \mathcal{A}_{nz}$, τ_0 will be constrained to lie in an interval whose limits are a function of the state and determined by

$$\max_{i \in I(x_1)} \left\{ -\frac{b_i(\mathbf{x})}{a_i(x_1)} + \frac{\underline{\tau_i}(\mathbf{x})}{|a_i(x_1)|} \right\} \leq \tau_0 \leq \min_{i \in I(x_1)} \left\{ -\frac{b_i(\mathbf{x})}{a_i(x_1)} + \frac{\overline{\tau_i}(\mathbf{x})}{|a_i(x_1)|} \right\} .$$

Define

$$\mathcal{A} = \mathcal{A}_{nz} \cap \mathcal{A}_z \, .$$

Then we see that (2.9) can be solved for $\mathbf{x}$ if and only if $\mathbf{x} \in \mathcal{A}$. We will call $\mathcal{A}$ the *admissible region.*

Write

$$\mathcal{U}_0(\mathbf{x}) = \left[\max_{i \in I(x_1)} \left\{ -\frac{b_i(\mathbf{x})}{a_i(x_1)} + \frac{\underline{\tau_i}(\mathbf{x})}{|a_i(x_1)|} \right\}, \min_{i \in I(x_1)} \left\{ -\frac{b_i(\mathbf{x})}{a_i(x_1)} + \frac{\overline{\tau_i}(\mathbf{x})}{|a_i(x_1)|} \right\} \right] .$$

Then we see that $\mathcal{U}_0(\mathbf{x}) = \emptyset$ if $\mathbf{x} \notin \mathcal{A}$ and $\mathcal{U}_0(\mathbf{x}) \neq \emptyset$ if $\mathbf{x} \in \mathcal{A}$. We will call $\mathcal{U}_0(\mathbf{x})$ the interval of *admissible control values* when in the state $\mathbf{x}$.

The original optimal control problem is now equivalent to:

Find the control $\tau_0(t)$ which minimises the performance measure

$$J(\mathbf{x}(0)) = \int_0^{t_f} L(\mathbf{x}(t), \tau_0(t))\, dt\,, \tag{2.11}$$

subject to the dynamics

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tau_0\,, \tag{2.12}$$

to the state and control constraints

$$\mathbf{x}(t) \in \mathcal{A}, \quad \tau_0(t) \in \mathcal{U}_0(\mathbf{x}(t)), \quad 0 \leq t \leq t_f\,,$$

where $\mathcal{A}$ and $\mathcal{U}_0(\mathbf{x})$ are defined above, and to the boundary conditions $\mathbf{x}_0 = \mathbf{x}(0)$ and $\mathbf{x}_f = \mathbf{x}(t_f)$.

Thus, we have reduced the problem of the simultaneous solution of n second order differential equations, to that of the solution of one second order differential equation which is subject to state and control constraints.

2.4 Solution of the Equivalent Time-Optimal Control Problem

We use dynamic programming to solve the time-optimal control problem of §2.3. Dynamic programming is a very general and powerful methodology and can readily handle state and control constraints, and non-trivial admissible regions [60] as well as various discontinuities. Use of dynamic programming entails the definition of a value function (called the *minimum time function* if the problem is purely minimum-time control). This function satisfies the *Hamilton-Jacobi-Bellman* (HJB) equation, a nonlinear partial differential equation (PDE). Assuming that the value function is sufficiently

smooth, the optimal control policy can be determined from the HJB equation, via the verification theorem [30].

We define the value function, $T(\mathbf{x})$, by

$$T(\mathbf{x}) = \inf_{\tau_0 \in \mathcal{U}_0(\mathbf{x})} \left\{ \int_0^{t_f} L(\mathbf{x}(t), \tau_0(t))\, dt \;:\; \mathbf{x}(0) = \mathbf{x},\ \mathbf{x}(t_f) = \mathbf{x}_f \right\},$$

where $\mathbf{x}$ is the solution of (2.12) and $\mathcal{U}_0(\mathbf{x})$ is the interval of admissible controls defined above. Notice that $T(\mathbf{x}) \in [0, +\infty]$ and $T(\mathbf{x}) = +\infty$ if $\mathbf{x} \notin \mathcal{A}$ since $\mathcal{U}_0(\mathbf{x}) = \emptyset$. Also, $T(\mathbf{x}) < +\infty$ if $\mathbf{x}$ is controllable to $\mathbf{x}_f$, otherwise $T(\mathbf{x}) = +\infty$. Let $\mathcal{C}$ denote the set of points controllable to $\mathbf{x}_f$. Then $\mathcal{C} \subseteq \mathcal{A}$.

The Hamilton-Jacobi-Bellman equation for the optimal-control problem is

$$\begin{cases} 0 = \min\limits_{\tau_0 \in \mathcal{U}_0(\mathbf{x})} \{\nabla T(\mathbf{x}) \cdot \mathbf{b}(\mathbf{x}, \tau_0) + L(\mathbf{x}, \tau_0)\}, \ \mathbf{x} \in \mathcal{C} \\ \\ T(\mathbf{x}_f) = 0 \\ 0 \le T(\mathbf{x}) \le 1 \quad \text{if } \mathbf{x} \ne \mathbf{x}_f \\ T(\mathbf{x}) = +\infty \quad \text{if } \mathbf{x} \in \partial\mathcal{C} \end{cases}$$

where $\mathbf{b}(\mathbf{x}, \tau_0)$ represents the system dynamics (2.12). The boundary conditions for this PDE are $S(\mathbf{x}_f) = 0$ at the target state $\mathbf{x}_f$ and $S(\mathbf{x}) = +\infty$ for $\mathbf{x}$ on the boundary of the controllable subspace, $\partial\mathcal{C}$.

Because $T(\mathbf{x}) = +\infty$ is difficult to deal with numerically, we consider the *discounted* value function (see Bardi and Falcone [5] and Evans and James [29] for the case $L(\mathbf{x}(t), \tau_0(t)) = 1$).

$$S(\mathbf{x}) = \inf_{\tau_0 \in \mathcal{U}_0(\mathbf{x})} \left\{ \int_0^{t_f} e^{-\lambda \int_0^t L(\mathbf{x}(s), \tau_0(s), s)\, ds} \lambda L(\mathbf{x}(t), \tau_0(t), t)\, dt \;:\; \mathbf{x}(0) = \mathbf{x} \quad \mathbf{x}(t_f) = \mathbf{x}_f \right\},$$

whence

$$S(\mathbf{x}) = 1 - e^{-\lambda T(\mathbf{x})}, \tag{2.13}$$

and $S(\mathbf{x}) \in [0, 1]$. Note that this transformation improves the numerical conditioning of the problem by reducing the range of the variables.

The HJB equation for the discounted optimal-control problem is

$$\begin{cases} 0 = \min\limits_{\tau_0 \in \mathcal{U}_0(\mathbf{x})} \{\nabla S(\mathbf{x}) \cdot \mathbf{b}(\mathbf{x}, \tau_0) + \lambda L(\mathbf{x}, \tau_0)(1 - S(\mathbf{x}))\}, \ \mathbf{x} \in \mathcal{C} \\ S(\mathbf{x}_f) = 0 \\ 0 \leq S(\mathbf{x}) \leq 1 \quad \text{if } \mathbf{x} \neq \mathbf{x}_f \\ S(\mathbf{x}) = 1 \qquad \text{if } \mathbf{x} \in \partial\mathcal{C} \end{cases} \tag{2.14}$$

The boundary conditions for this PDE are $S(\mathbf{x}_f) = 0$ at the target state $\mathbf{x}_f$ and $S(\mathbf{x}) = 1$ for $\mathbf{x}$ on the boundary of the controllable subspace, $\partial\mathcal{C}$. Note that since the transformation (2.13) is monotonic, the minimising control τ_0 is the same for both the un-discounted and discounted problems.

If the HJB equation (2.14) has a smooth solution $S(\cdot)$, then the verification theorem of optimal control [30] implies that if $\tau_0^*(\mathbf{x})$ achieves the infimum in (2.14), then $\tau_0^*(\mathbf{x})$ is an optimal feedback control, if sufficiently regular. However, in general, (2.14) does not have a smooth solution. In fact, typically $S(\cdot)$ is only Hölder continuous with exponent $0 < \alpha < 1$ [5, 29] (such functions are not everywhere differentiable), and so does not satisfy (2.14) in a classical sense, but only in a weak sense, viz. the viscosity sense [31]. Because we cannot solve (2.14) explicitly, we must resort to a numerical approximation $S^h(\cdot)$ of $S(\cdot)$. From this, an approximate optimal policy, $\tau_0^{*h}(\mathbf{x})$, will be obtained.

Let $\mathcal{D}^h$ be a rectangular grid of size $h > 0$ covering the admissible region $\mathcal{A}$, or the

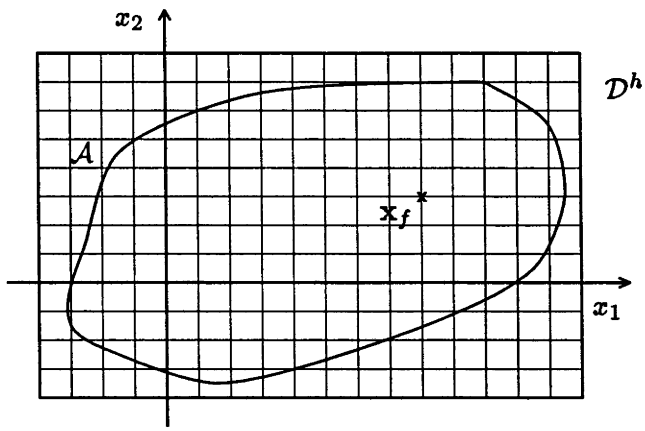


Figure 2.2: The State Space $\mathcal{D}^h$

part of $\mathcal{A}$ of interest, Figure 2.2. Now, approximate $\nabla S(\mathbf{x})$ by a first order approximation over a grid of size h, so that

$$\nabla S(\mathbf{x}) \cdot \mathbf{b}(\mathbf{x}, \tau_0) \approx \sum_{i=1}^{2} \left\{ \frac{S^h(\mathbf{x} \pm h e_i) - S^h(\mathbf{x})}{h} b_i^{\pm}(\mathbf{x}, \tau_0) \right\} \tag{2.15}$$

where $e_1 = (1,0)^T$ and $e_2 = (0,1)^T$, the $\pm$ notation means $b_i^{\pm} = +b_i$ if $b_i \geq 0$ and $b_i^{\pm} = -b_i$ if $b_i \leq 0$, $S(\mathbf{x} + h e_i)$ is taken when $b_i \geq 0$ and $S(\mathbf{x} - h e_i)$ when $b_i < 0$. Note that the difference quotient respects the system dynamics. This leads to algorithms with good stability properties, Kushner and Dupuis [41].

If we now define

$$\mathbf{v}(\mathbf{x}) \triangleq \max_{\tau_0 \in \mathcal{U}_0(\mathbf{x})} \left\{ \sum_{i=1}^{2} |b_i(\mathbf{x}, \tau_0)| \right\}, \qquad \Delta t^h(\mathbf{x}) \triangleq \frac{h}{\mathbf{v}(\mathbf{x})}, \tag{2.16}$$

noting that in our formulation $\mathbf{v}(\mathbf{x})$ is simply

$$\mathbf{v}(\mathbf{x}) \triangleq \max_{\tau_0 \in \mathcal{U}_0(\mathbf{x})} \{|x_2| + |\tau_0|\} = |x_2| + \max\{|\underline{\tau_0}(\mathbf{x})|, |\overline{\tau_0}(\mathbf{x})|\} ,$$

then using a modification of an algorithm presented by Kushner and Dupuis [41] (Chapter 4, pp 76-77), the HJB equation (2.14) may be re-written in the form

$$\begin{cases} S^h(\mathbf{x}) = \min\limits_{\tau_0 \in \mathcal{U}_0(\mathbf{x})} \left\{ \dfrac{1}{1 + \lambda L(\mathbf{x}, \tau_0) \Delta t^h(\mathbf{x})} \left(\sum\limits_{z \in \mathcal{N}^h(\mathbf{x})} P^h(z|\mathbf{x}, \tau_0) S^h(z) + \lambda L(\mathbf{x}, \tau_0) \Delta t^h(\mathbf{x}) \right) \right\}, \\ \qquad \mathbf{x} \in \operatorname{int} \mathcal{D}^h \\ S^h(\mathbf{x}_f) = 0 \\ 0 \leq S^h(\mathbf{x}) \leq 1 \quad \text{if } \mathbf{x} \neq \mathbf{x}_f \\ S^h(\mathbf{x}) = 1 \qquad \text{if } \mathcal{U}_0(\mathbf{x}) = \emptyset \ (\mathbf{x} \notin \mathcal{A}), \text{ or } \mathbf{x} \in \partial \mathcal{D}^h \end{cases} \tag{2.17}$$

where $\mathcal{N}^h(\mathbf{x}) = \{\mathbf{x}, \mathbf{x} \pm h e_i\}$ is the simplex consisting of $\mathbf{x}$ and its nearest neighbours in the discretised state space, $\mathcal{D}^h$,

$$P^h(z|\mathbf{x}, \tau_0) = \begin{cases} \dfrac{b_i^{\pm}(\mathbf{x}, \tau_0)}{\mathbf{v}(\mathbf{x})} & \text{if } z = \mathbf{x} \pm h e_i \\ 1 - \dfrac{|b_1(\mathbf{x}, \tau_0)| + |b_2(\mathbf{x}, \tau_0)|}{\mathbf{v}(\mathbf{x})} & \text{if } z = \mathbf{x} \\ 0 & \text{otherwise} \end{cases}$$

and we note that $\sum_{z\in\mathcal{N}^h(\mathbf{x})} P^h(z|\mathbf{x},\tau_0) = 1$. The numbers $P^h(z|\mathbf{x},\tau_0)$ can be interpreted as transition probabilities for a controlled Markov chain. Kushner and Dupuis show that Markov chain approximation methods produce numerical schemes which have nice convergence properties, under broad conditions.

Let $S^h(\mathbf{x})$, $\mathbf{x}\in\mathcal{D}^h$, be the solution of (2.17). Then $\tau_0^{*h}(\mathbf{x}) \in \mathcal{U}_0(\mathbf{x})$ minimising the right hand side of the PDE in (2.17) is the desired approximation to $\tau_0^*(\mathbf{x})$.

There are two classical methods of solving the boundary value problem (2.17), namely the *value space* or *Jacobi* method, which can be interpreted as a fixed point iteration method, and the *policy space* method which is analogous to a gradient procedure in the space of controls, [41]. There are also many variations thereof, mostly aimed at speeding up the rate of convergence. More discussion of computational issues will be given in §2.6 after presentation of the following examples.

2.5 Examples

We now apply the theory developed in §2.2, §2.3 and §2.4 to examples based on an experimental manipulator that we have in our laboratory. The manipulator is a 4 degree-of-freedom SCARA arm and is described in Appendix A. For simplicity, we restrict the motion to be planar, driven by only the first and second joint motors.

The planar dynamics of the SCARA are modelled as

$$\hat{\mathbf{M}}(\mathbf{q})\ddot{\mathbf{q}} + \hat{\mathbf{v}}(\mathbf{q},\dot{\mathbf{q}}) = \boldsymbol{\tau} \tag{2.18}$$

where $\hat{\mathbf{M}}(\mathbf{q})$ and $\hat{\mathbf{v}}(\mathbf{q},\dot{\mathbf{q}})$ represent the mass matrix and coriolis-centrifugal vector respectively, and are defined in equations (A.2) and (A.3) of Appendix A.

The parameter values for $\hat{\mathbf{M}}(\mathbf{q})$ and $\hat{\mathbf{v}}(\mathbf{q},\dot{\mathbf{q}})$, identified using the standard least squares technique, are shown in Table 2.1.

Parameter	Value	Units	Represents
f_{11}	0.3974	Nms^2/rad	Inertia terms
$f_{12} = f_{21}$	0.1987	Nms^2/rad	
g_{11}	3.7694	Nms^2/rad	
$g_{12} = g_{21}$	1.8847	Nms^2/rad	
h_{11}	37.9687	Nms^2/rad	
$h_{12} = h_{21}$	2.9851	Nms^2/rad	
h_{22}	17.2015	Nms^2/rad	
v_{1s}	-3.7694	Nms^2/rad^2	Centrifugal
v_{2s}	1.8847	Nms^2/rad^2	and coriolis
w_{1s}	-1.8847	Nms^2/rad^2	terms
v_{1c}	0.3974	Nms^2/rad^2	
v_{2c}	-0.1987	Nms^2/rad^2	
w_{1c}	0.1987	Nms^2/rad^2	
l_1	0.6100	m	Length of links
l_2	0.5800	m	1 and 2

Table 2.1: SCARA Manipulator Parameter Data

The torque limits for this robot are given by

$$\left.\begin{array}{rcl} \underline{\tau}(\dot{\mathbf{q}}) & = & \max \{-309.42, -1146.0 - 984.987\dot{\mathbf{q}}\} \\ \overline{\tau}(\dot{\mathbf{q}}) & = & \min \{+309.42, +1146.0 - 984.987\dot{\mathbf{q}}\} \end{array}\right\}, \tag{2.19}$$

and are due to current and voltage limits of $6A$ and $40V$ respectively.

We choose the following joint space path parameterisation which traces a circle in the real physical space, Figure 2.3:

$$\mathbf{q} = \begin{pmatrix} q_1 \\ q_2 \end{pmatrix} = \begin{pmatrix} \arctan 2\left(\left(\frac{-\alpha y_1(s)+\beta y_2(s)}{\alpha^2+\beta^2}\right), \left(\frac{\beta y_1(s)+\alpha y_2(s)}{\alpha^2+\beta^2}\right)\right) \\ \arccos\left(\frac{y_1^2(s)+y_2^2(s)-l_1^2-l_2^2}{2l_1l_2}\right) \end{pmatrix} = \mathbf{f}(s)\,, \tag{2.20}$$

where

$$\begin{aligned} y_1(s) &= 0.55 - \sqrt{0.125}\cos(s)\,, \\ y_2(s) &= 0.55 + \sqrt{0.125}\sin(s)\,, \\ \alpha &= l_2\sin(q_2)\,, \\ \beta &= l_1 + l_2\cos(q_2)\,, \end{aligned}$$

$s \in [0, 6.3]$, and where $\sqrt{0.125}$ is the radius of the circle, and $(0.55, 0.55)^T$ its centre $(6.3 \approx 2\pi)$.

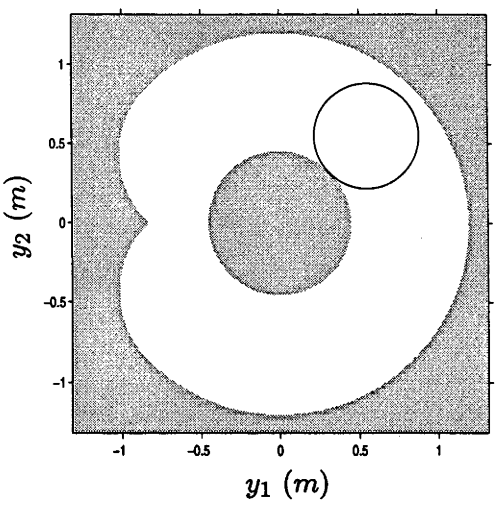


Figure 2.3: The Cartesian Space Path

Having specified the dynamics of the manipulator, (2.18), the torque limits, (2.19), and the joint path parameterisation, (2.20), we can now calculate the admissible region $\mathcal{A}$ and admissible control space $\mathcal{U}_0(\mathbf{x})$. Choosing $h = 0.0125$, these are displayed in Figures 2.4 and 2.5 respectively.

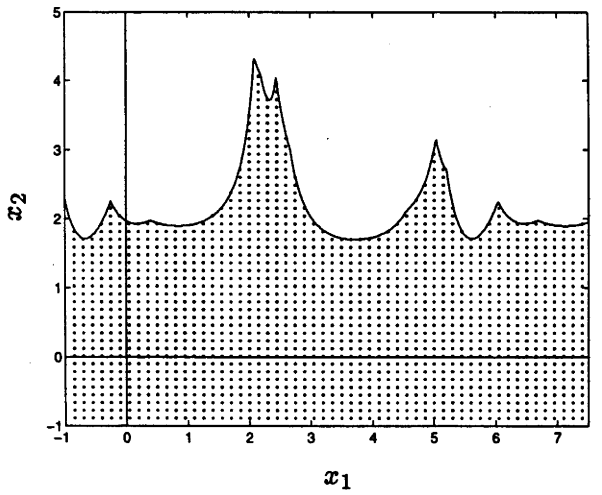


Figure 2.4: Admissible Region - $\mathcal{A}$

We now consider two different optimality criteria. The first criterion is a pure minimum-time cost and allows us to compare the solution using dynamic programming

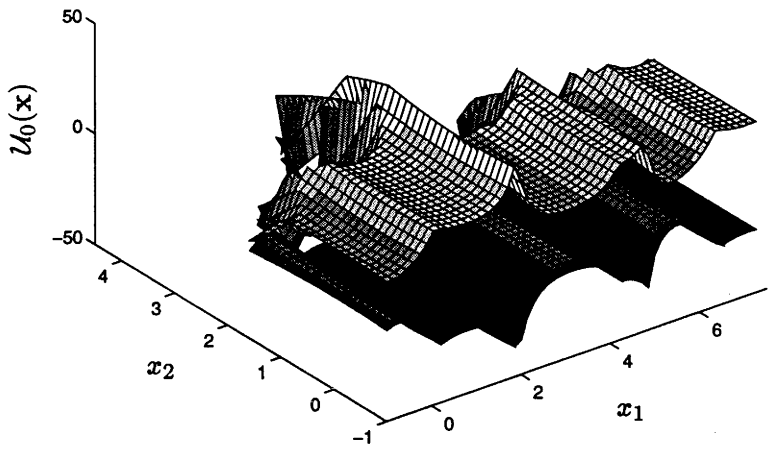


Figure 2.5: Admissible Controls - $\mathcal{U}_0(\mathbf{x})$

versus the equivalent solution calculated using a PMP based shooting method. The second criterion is a minimum-time plus quadratic state and control cost which serves to demonstrate the versatility of the dynamic programming approach when calculating solutions which are not bang-bang in nature.

2.5.1 Pure Minimum-Time Example

In the pure minimum-time case, we set $L(\mathbf{x}(t), \tau_0(t)) = 1$ which provides the minimum-time function $T(\mathbf{x}) = \int_0^{t_f} 1dt = t_f$, as required. The numerical solution of the corresponding HJB equation (2.17) then yields the discretised discounted minimum-time function $S^h(\mathbf{x})$, from which the minimum-time can be recovered via

$$t_f = -\frac{1}{\lambda} \log_e\{1 - S^h(\mathbf{x})\} .$$

Solution of the Example

Solution of the derived HJB equation (2.17), choosing $\lambda = 1$ for simplicity and using a desired final state $\mathbf{x}_f = (6.3, 0)^T$, yields the discretised discounted minimum-time

function $S^h(\mathbf{x})$ and the optimal control policy $\tau_0^{*h}(\mathbf{x})$ shown in Figures 2.6 and 2.7. Note from Figure 2.6 that in this example, there are no points in the interior of $\mathcal{A}$ where $S^h(\mathbf{x}) = 1$. Thus, the controllable subspace $\mathcal{C}$ coincides with the admissible region $\mathcal{A}$. (Other examples exist where $S^h(\mathbf{x})$ is close to 1 in the interior of $\mathcal{A}$. Such regions represent those states which are uncontrollable to the target state $\mathbf{x}_f$.)

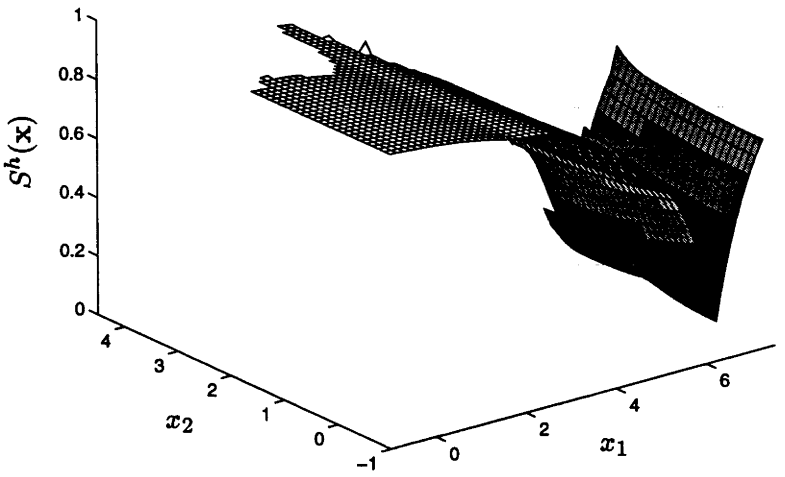


Figure 2.6: Value Function - $S^h(\mathbf{x})$

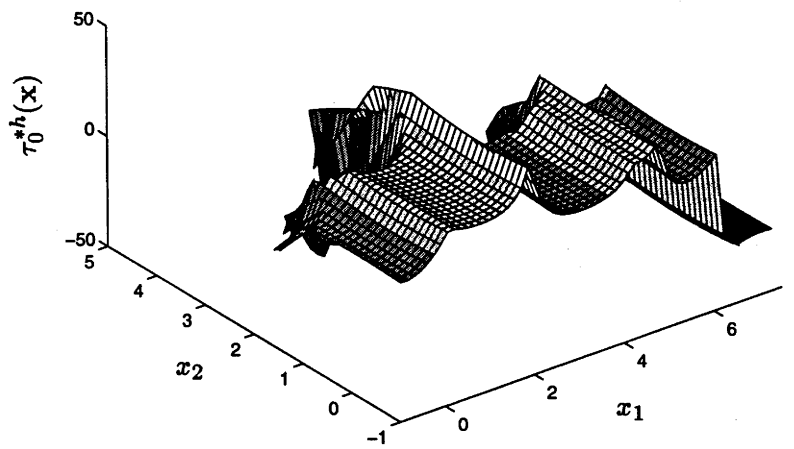


Figure 2.7: Optimal Control Policy - $\tau_0^{*h}(\mathbf{x})$

Using the initial state $\mathbf{x}(0) = \mathbf{0}$, the discounted minimum-time function provides $S^h(\mathbf{x}) \approx 0.976$ which predicts the minimum-time to complete the task as $t_f \approx 3.746s$.

We now integrate equations (2.12), choosing an initial state of $\mathbf{x}(0) = \mathbf{0}$ and applying the optimal feedback control policy $\tau_0^{*h}(\mathbf{x})$ at each integration step by feeding back the state information. This yields the time-optimal path trajectories $\mathbf{x}^*(t)$, displayed in the phase plane in Figure 2.8. The control $\tau_0^*(t)$ achieving these trajectories, and the corresponding controls $\tau_1^*(t)$ and $\tau_2^*(t)$ are shown in Figure 2.9. Note that the time taken to reach the desired final state $\mathbf{x}_f$ is $t_f \approx 3.727s$ which is marginally less than that predicted by the discounted minimum-time function.

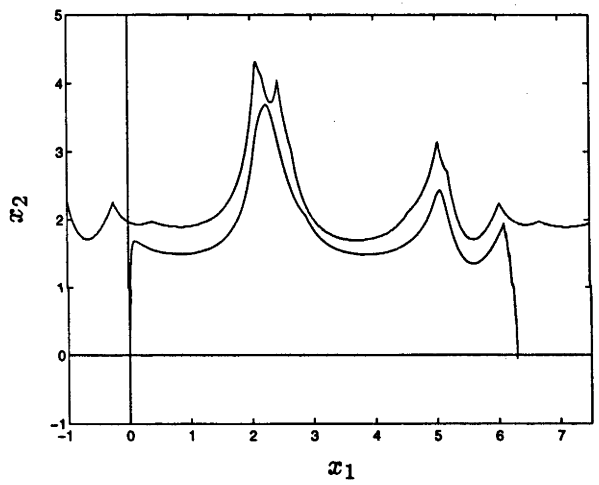


Figure 2.8: Optimal Path Trajectory - $x_2^*(x_1^*)$

The time taken for the system to reach the desired final state compares well with the minimum-time solution $t_f \approx 3.683s$ calculated using the PMP based shooting method of Bobrow *et al.* [8]. The estimate is in error by only approximately 1%, which shows that the numerical methods work well.

Finally, the time-optimal path trajectories $\mathbf{x}^*(t)$ are fed into equation (2.20) and its derivatives to produce the optimal joint trajectories $\mathbf{q}(t)$, $\dot{\mathbf{q}}(t)$ and $\ddot{\mathbf{q}}(t)$, Figure 2.10.

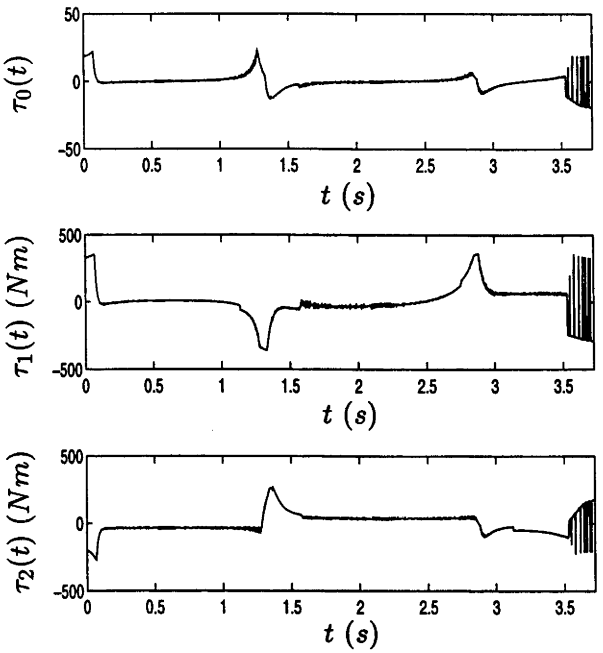


Figure 2.9: Optimal Path Control - $\tau_i^*(t)$=$\tau_i^*(\mathbf{x}^*(t))$

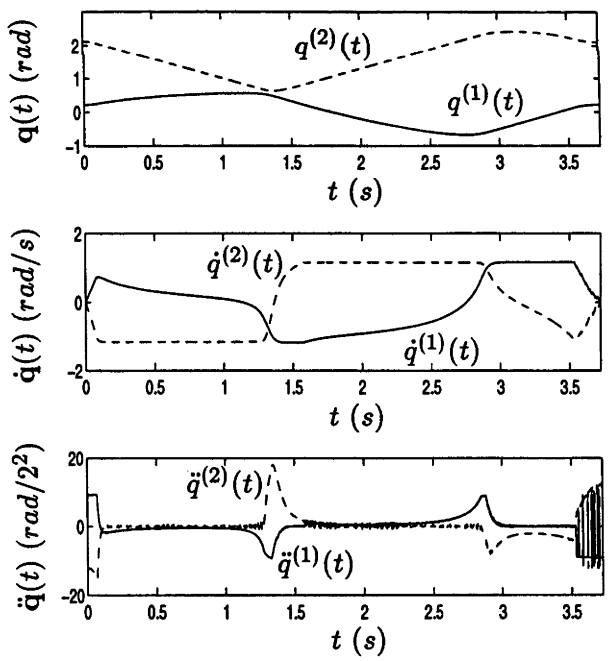


Figure 2.10: Joint Trajectories - $\mathbf{q}(t)$,$\dot{\mathbf{q}}(t)$,$\ddot{\mathbf{q}}(t)$

Application of the Result

To implement this solution on our experimental SCARA manipulator, we use a computed-torque controller, which is a model based-controller and seems an appropriate choice given that the reference trajectory which will drive the robot is calculated based on the model.

The controller gains are chosen as $\mathbf{k}_p = (100, 100)^T$ and $\mathbf{k}_v = (20, 20)^T$, and correspond to critical damping and natural frequencies of 10 rad/s (similar figures are commonly used in the literature).

The time-optimal joint trajectories, Figure 2.10, are applied as reference to the experimental system and the results logged. Figure 2.11 displays the desired Cartesian space path of the end-effector, shown as the solid line, and the actual path traced by the end-effector, shown as the dashed line. The tracking appears to be quite good. However, if we examine a plot of the joint angle errors, Figure 2.12, we see that the error is $\sim O(0.05\ rad)$ which through the forward kinematics equates to approximately $5cm$ in the end-effector position.

Figure 2.13 displays the levels of commanded torques that are clipped when the actuators saturate. Comparing this to Figure 2.12, it is apparent that the joint angle errors, and hence the end-effector tracking error grow as the actuators saturate. This is intuitively obvious since this is when the ability to control to reject the errors is lost.

In Figure 2.14, we see that the torques demanded of the actuators contain some high frequency noise. Early in the trajectory the noise is not overly large, but later on there are rapid oscillations in the signal. A priori, our concern was that this would manifest itself as high frequency oscillations during the experiment. In fact, such oscillations were not visible during the experiment.

These results will be discussed in more detail in §2.6.

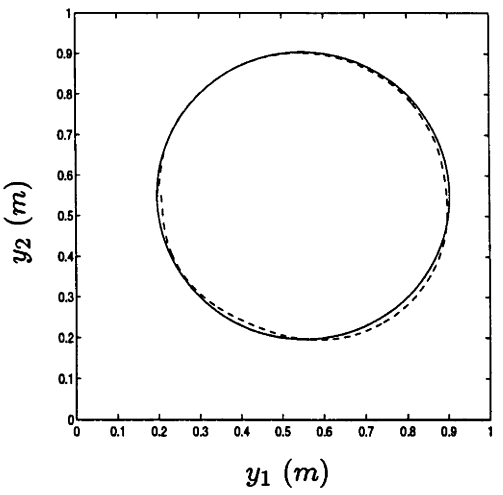


Figure 2.11: Cartesian Space Path - Predicted and Actual

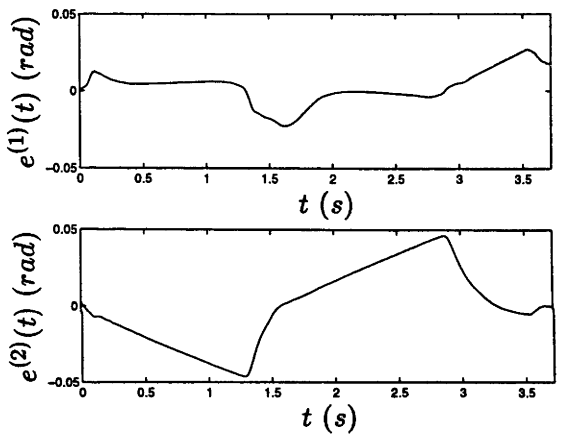


Figure 2.12: End-Effector Tracking Error

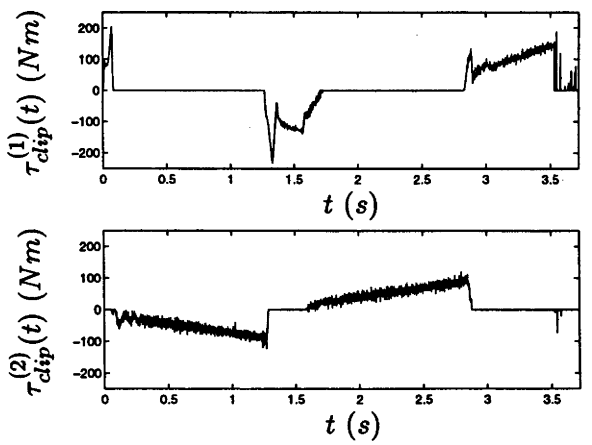


Figure 2.13: Torque Clipping

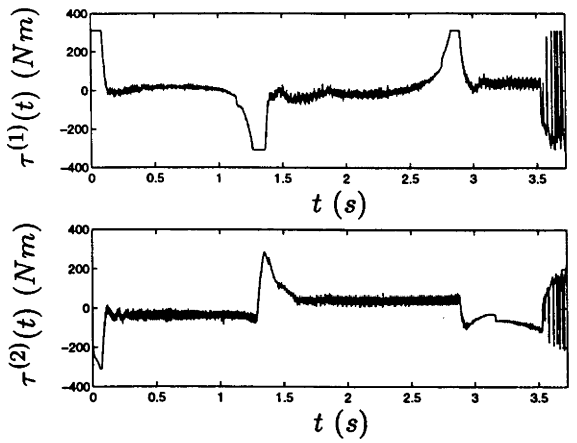


Figure 2.14: Applied Torques

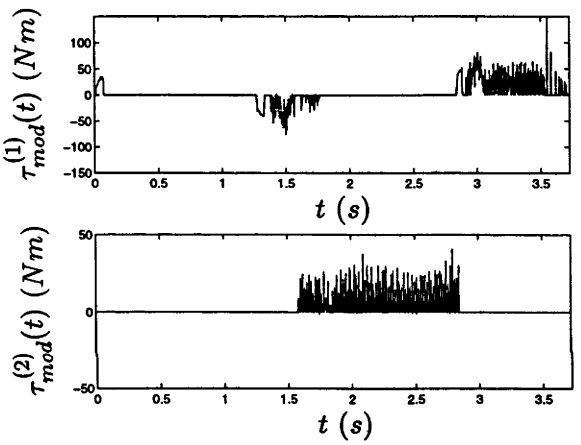


Figure 2.15: Difference in Model Based Control for Numerical and Online Data

2.5.2 Minimum-Time Plus Quadratic Cost Example

The second criterion that we consider is a minimum-time plus quadratic state and control cost of the form

$$L(\mathbf{x}(t), \tau_0(t)) = 1 + 0.25x_2^2 + 0.02\tau_0^2 \, .$$

This example serves to demonstrate the versatility of the dynamic programming approach when calculating solutions which are not bang-bang in nature.

The anticipated effect of the additional penalty terms is that the term in x_2 will slow down the trajectory at all points along the path whilst the term in τ_0 will do

likewise, and also smooth out τ_0, and hence the joint torques, at transitions (remember that $\tau_0 \equiv \ddot{s}$ and so we are limiting the acceleration).

Note that this in this example, $S^h(\mathbf{x})$ is not the discounted minimum-time function. Thus, we cannot determine the time taken to traverse the path directly from $S^h(\mathbf{x})$.

Solution of the Example

Solution of the derived HJB equation (2.17), again choosing $\lambda = 1$ for simplicity and using the same target state $\mathbf{x}_f = (6.3, 0)^T$, yields the value function $S^h(\mathbf{x})$ and optimal control policy $\tau_0^{*h}(\mathbf{x})$ shown in Figures 2.16 and 2.17 respectively . Notice in the optimal control policy the smoothing effect of the penalty on the control.

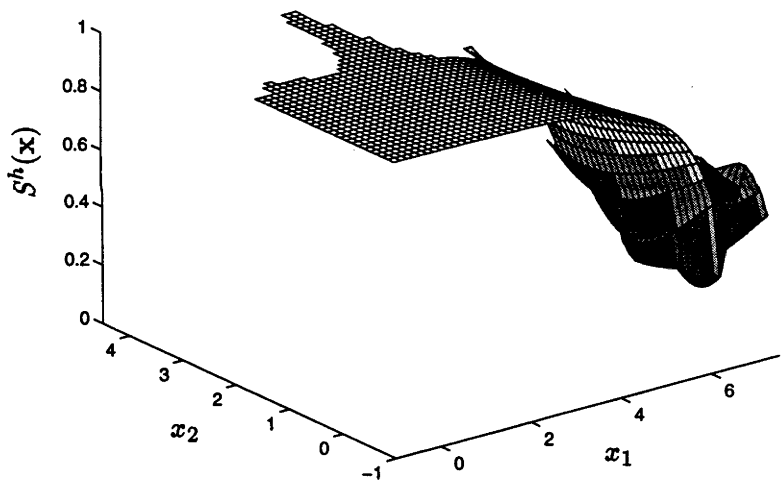


Figure 2.16: Value Function - $S^h(\mathbf{x})$

The optimal path trajectory, $\mathbf{x}^*(t)$, is shown in the phase plane Figure 2.18. The control $\tau_0^*(t)$ achieving this, and the corresponding controls $\tau_1^*(t)$ and $\tau_2^*(t)$ are shown in Figure 2.19. The optimal joint trajectories $\mathbf{q}(t)$, $\dot{\mathbf{q}}(t)$ and $\ddot{\mathbf{q}}(t)$ are shown in Figure 2.20. Note the effect of the penalty on x_2 causing the suppression of the trajectory (*c.f.* the pure minimum-time example, Figure 2.8). Note also the effect of the penalty on τ_0

causing much smoother transitions in the control signals (*c.f.* Figure 2.9). Note finally the resulting increase in the time to complete the task compared to the pure minimum-time case, from $3.723s$ to $4.300s$.

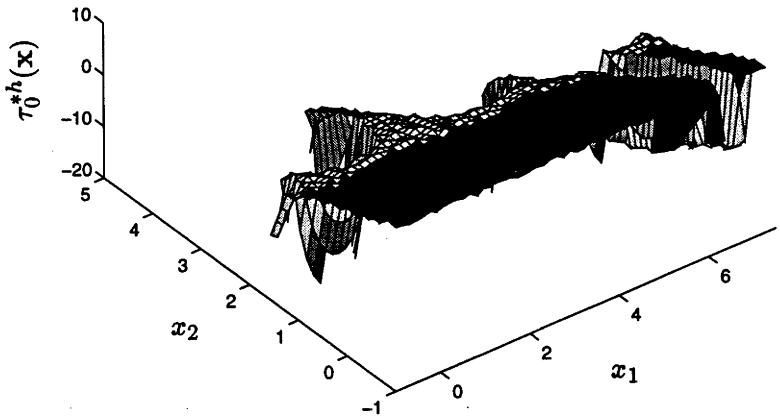


Figure 2.17: Optimal Control Policy - $\tau_0^{*h}(\mathbf{x})$

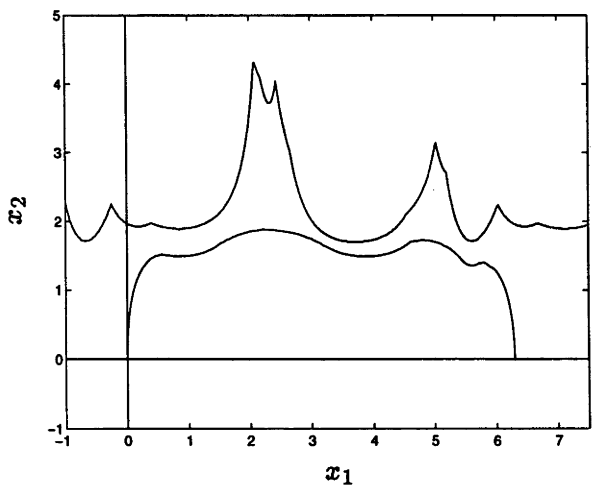


Figure 2.18: Optimal Path Trajectory - $x_2^*(x_1^*)$

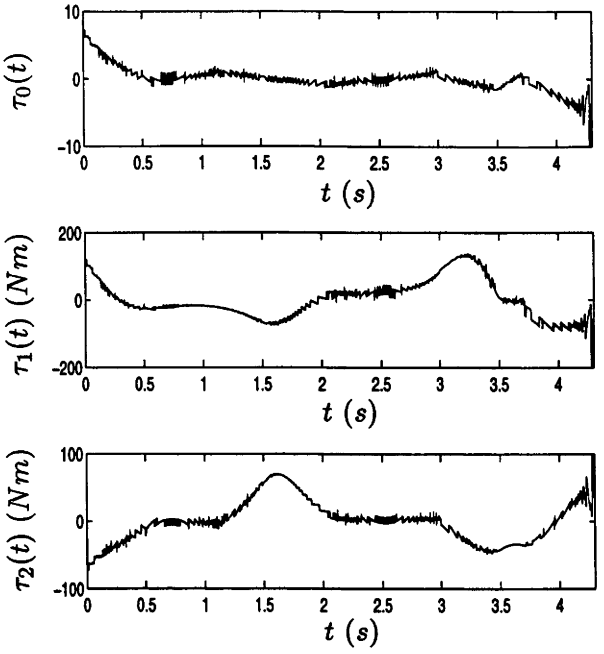


Figure 2.19: Optimal Path Control - $\tau_i^*(t)$=$\tau_i^*(\mathbf{x}^*(t))$

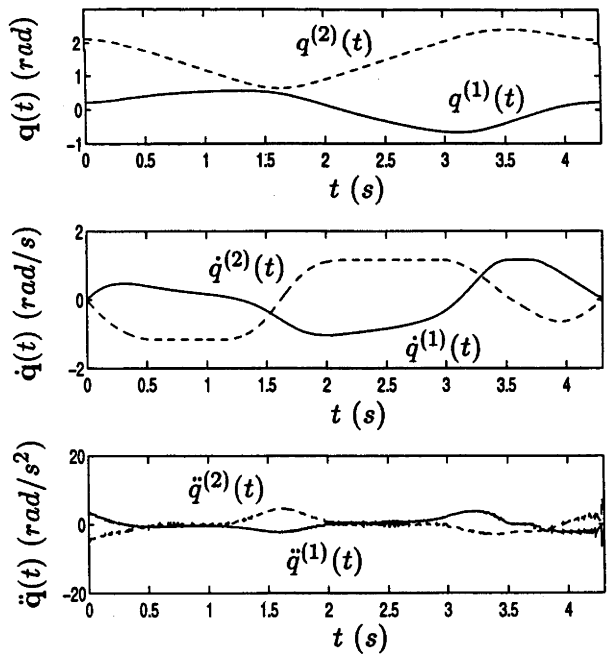


Figure 2.20: Joint Trajectories - $\mathbf{q}(t)$,$\dot{\mathbf{q}}(t)$,$\ddot{\mathbf{q}}(t)$

Application of the Result

We now implement the reference trajectory on our experimental SCARA manipulator, using the same computed-torque controller and gain settings as before.

Figure 2.21 displays the desired Cartesian space path of the end-effector, shown as the solid line, and the actual path traced by the end-effector, shown as the dashed line. Clearly, the tracking is better than for the pure minimum-time example. Indeed, if we examine a plot of the joint angle errors, Figure 2.22, we see that the error is now only $\sim O(0.03\ rad)$, which through the forward kinematics equates to only approximately $3cm$ in the end-effector position.

Figure 2.23 displays the levels of commanded torques that are clipped when the actuators saturate. It is again evident, by comparison to Figure 2.22, that the errors are correlated to the clipping of the commanded torques. However, we note that the levels of clipping are less in this case than for the pure minimum-time example, *c.f.* Figure 2.13, and so although there is no more ability to control the errors (this ability is lost as soon as the actuator saturates), the difference driving the errors is less.

Note in Figure 2.24, that the level of noise present in the torques demanded of the actuators is much less than in the pure minimum-time example.

These results will be discussed in more detail in §2.6.

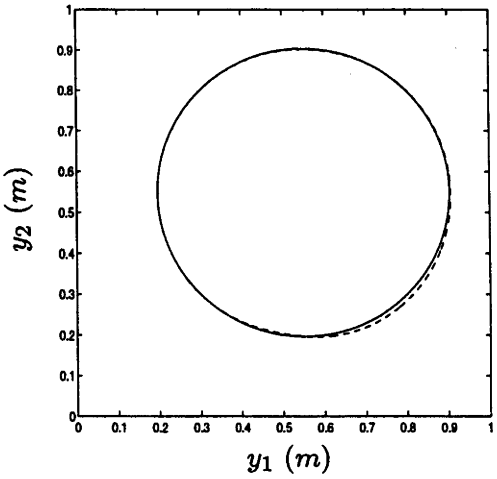


Figure 2.21: Cartesian Space Path - Predicted and Actual

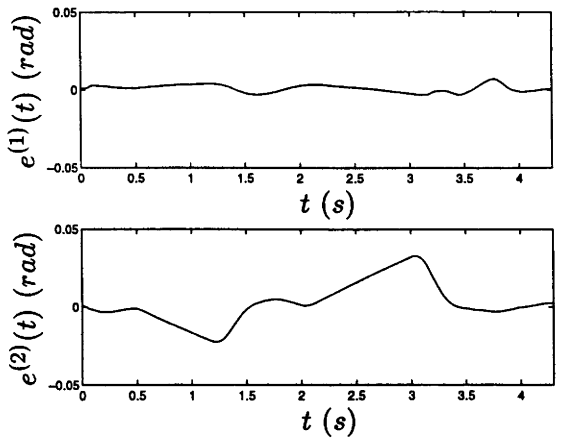


Figure 2.22: End-Effector Tracking Error

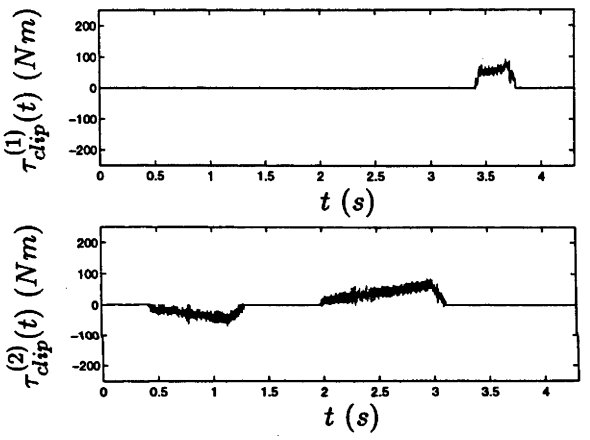


Figure 2.23: Torque Clipping

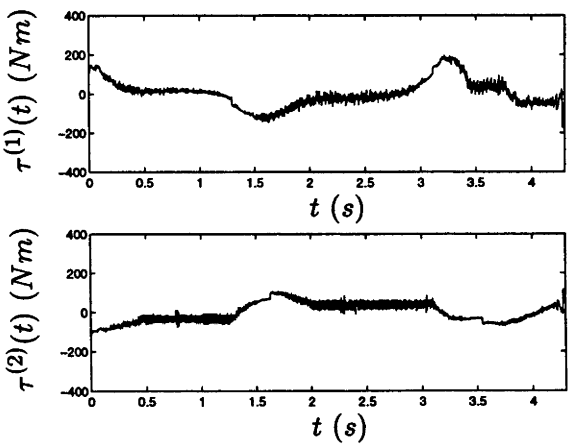


Figure 2.24: Applied Torques

2.6 Computational Issues

A major issue with the dynamic programming method is computational cost, viz. the memory requirements and the time required for computations. This depends critically on the state dimension n: the computational cost grows exponentially with n (the curse of dimensionality). Since in this problem the dimension is low (i.e. 2), computation is feasible in practice, as we have demonstrated. However, as with any numerical approximation scheme, there is a trade off between computational effort and accuracy.

As mentioned in § 2.4 above, the discounted minimum-time function $S(\cdot)$ is generally not everywhere differentiable, rather only Hölder continuous with exponent α, $0 < \alpha < 1$. This lack of differentiability has important practical consequences. Without constraints, theory suggests $\alpha = \frac{1}{2}$, with rate of convergence and the error in the approximation $e(h) \sim O(h^{\frac{1}{2}})$, as the step size $h \downarrow 0$. Our results are consistent with this prediction, Figures 2.25 through 2.28. This is a rather slow rate of convergence, meaning that for the error $e(h)$ to be small, the step size h must be very small with a consequent increase in the memory requirements and computational time. Indeed, the number of iterations to converge appears to double as h is halved, while the time to converge appears to increase by a factor of 8, Figures 2.25 and 2.26. This dramatic increase in time is of course due to the to the number of points in the domain of calculation, and hence also memory requirements, increasing by a factor of 4 as h is halved. This slow rate of convergence is due to the target-point constraint.

A further problem is that one must exercise care in determining when the numerical scheme has converged. Our measure of convergence $\gamma(\cdot)$ compares the change in the value function relative to the value function at each iteration, viz

$$\gamma(k) = \max_{\mathbf{x} \in \mathcal{D}} \left\{ \frac{S_k^h(\mathbf{x}) - S_{k-1}^h(\mathbf{x})}{S_{k-1}^h(\mathbf{x})} \right\}$$

between each iteration. In the examples presented above, we required the convergence measure $\gamma(k) \sim O(10^{-8})$ before the value function stopped changing significantly. This is seen when examining the (log of the) error for the pure minimum-time example, where

convergence is slow until it increases rapidly at $\gamma(k) \approx 10^{-8}$, Figure 2.29. Unfortunately, the $\gamma(k)$ which signifies convergence is different for different examples, and would not be known prior to solving the equation.

Considering all of these issues, it is important to solve the problem only as accurately as required. If significant modelling errors, disturbances, etc., are anticipated then errors of $5 - 10\%$ might be seen as an acceptable engineering trade-off, and so h need not be taken so small in practice. Also, we note that these computations are done off-line.

However, if high accuracy is important, there are schemes available to speed up the rate of convergence. Details of many such schemes are given in [41]. In our work, we tried a common acceleration scheme discussed by Capuzzo-Dolcetta and Falcone [27], and Kushner and Dupuis [41]. In our theory, we specify two boundary conditions for solution of the PDE; on the boundaries of the admissible region $\partial\mathcal{A}$ and/or domain of approximation $\partial\mathcal{D}$, and at the target state $\mathbf{x}_f$. Unfortunately, the acceleration method failed due to the presence of the internal boundary condition at the target state $\mathbf{x}_f$, which is related to the Hölder continuity of the (discounted) minimum-time function. Without such a condition, substantial acceleration is possible, as advertised. (Note also that if the external boundary condition is removed, whilst the solution ultimately converges to that calculated with the external boundary condition applied, there is a substantial degradation in performance - it is not possible to remove the target state boundary condition.)

We also tried multi-grid methods, where a solution calculated over a grid of size h is interpolated into a new grid with $h_{new} < h$, and this then used as the initial guess for a finer, and hence more accurate approximation. In fact, these methods proved fruitful. Figure 2.30 displays the time savings when using multi-grid method which interpolates a solution from a grid of size h to one of size $\frac{h}{2}$. The asterisks denote the time taken for the solution to converge, in cpu mins, without using the multi-grid method, and the o's represents the time taken for the solution to converge by repeatedly applying the

multi-grid method, starting from $h = 0.1$.

We have demonstrated that for our examples, it is possible to apply the results derived using a dynamic programming approach to a real system with comparable success to those derived using a PMP based shooting method. However, there are some points to note.

- Figure 2.15 displays the difference between the model-based part of the control law, calculated using the reference joint data (and shown in Figure 2.9, and the maximum torque which the actuator can deliver and which is the same except that it is based on the measured joint data. This difference is an artefact of the numerical approximation, being due to applying controls taken at the nearest grid-point when integrating equations (2.12). Of course, the effect becomes less apparent as h becomes smaller. This is not a problem with PMP based shooting methods, because values can be calculated at the exact, rather than discrete, state.

- We noted in Figure 2.14 that in the pure minimum-time case, the input to the actuators contained some high frequency noise. This was not unexpected since similar noise characteristics were apparent in the joint acceleration trajectory which is used to calculated this input, Figure 2.10. Our concern, of course, was that this noise would manifest itself as high frequency oscillations in the robot during the experiments. In fact, such behaviour was not apparent.

  We have concluded that the oscillations did not occur because the frequencies of the oscillations, $O(100\ Hz)$, are much higher than those of the system, $O(10\ Hz)$. Of course, if the torque signal were to contain oscillations with frequencies approaching that of the system, then this would become a major concern.

  The noise is due to the time-optimal trajectory tracking the *switching curve*, Figure 2.31. In our example this effect is apparent only in the final deceleration phase of the time-optimal trajectory since, for much of its earlier history, the trajectory traverses the boundary of the admissible region where there is no free-

dom of control. (This is because of the high acceleration rates that our robot possesses.) However, the amount of noise and its frequency content is example specific, depending also on the grid size h and on the integration time step dt. It is feasible that, in certain cases, oscillations could occur whose frequencies approach those of the system. Of course, this is not a problem with PMP based shooting methods.

Clearly, adding a penalty on the control in the cost function acts to smooth out the transitions along the switching curve, alleviating the problem, although producing a sub-optimal policy. However, as was demonstrated, the non bang-bang nature of the sub-optimal control policy gives more authority for control and thus facilitates better tracking.

Ultimately, the successful application of the resulting trajectories, calculated using dynamic programming or PMP based methods, will depend not only on whether the result can be applied, but on whether the desired objective can be achieved. This will be affected by many factors which neither approach accounts for; for example the difference between the model and the true plant, the unmodelled controller dynamics, measurement errors, etc.

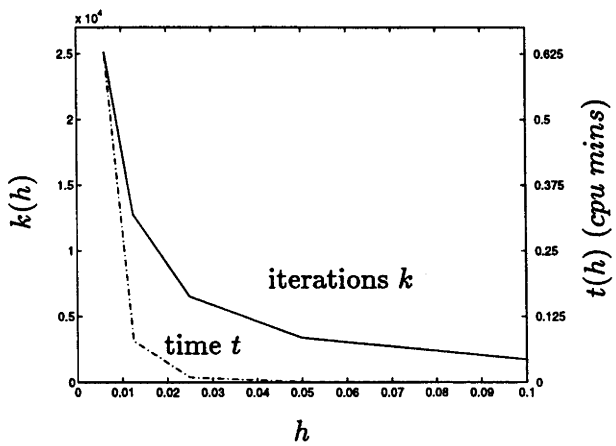


Figure 2.25: Time and Iterations to Converge versus h

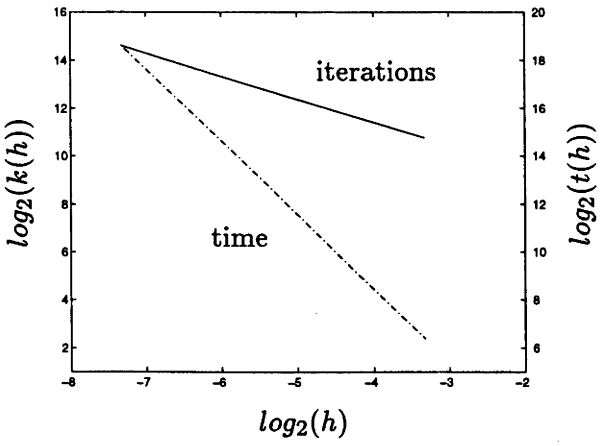


Figure 2.26: log_2(Time and Iterations to Converge) versus $log_2(h)$

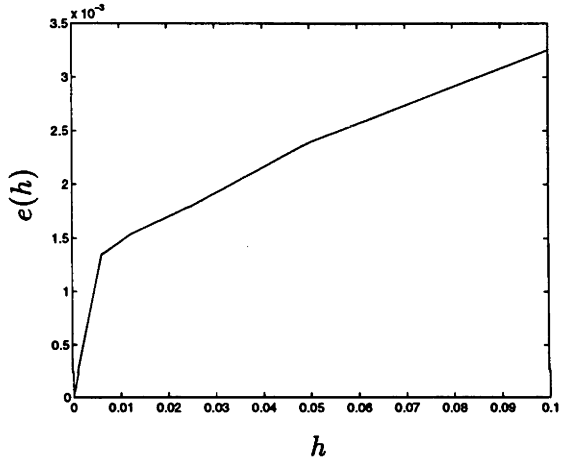


Figure 2.27: Error in $S^h(\cdot)$ versus h

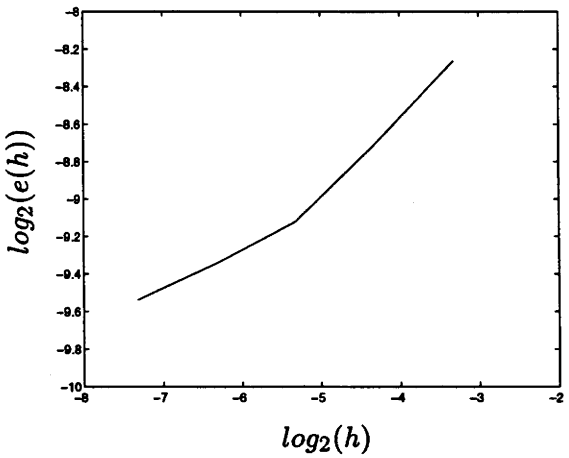


Figure 2.28: log_2(Error in $S^h(\cdot)$) versus $log_2(h)$

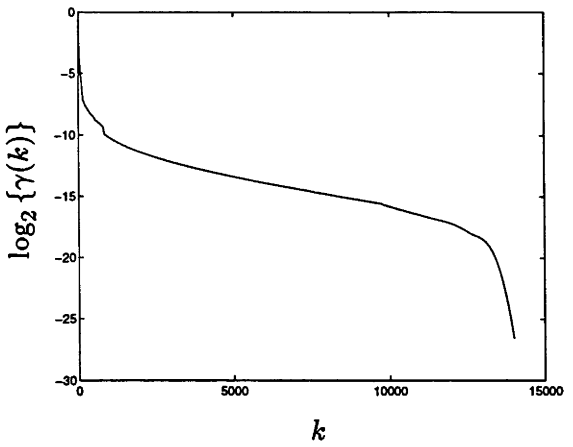


Figure 2.29: Identifying the Point of Convergence

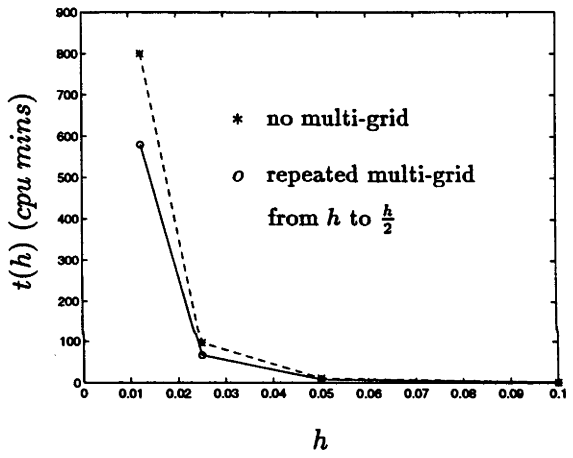


Figure 2.30: Time Saving Using Multi-Grid Technique

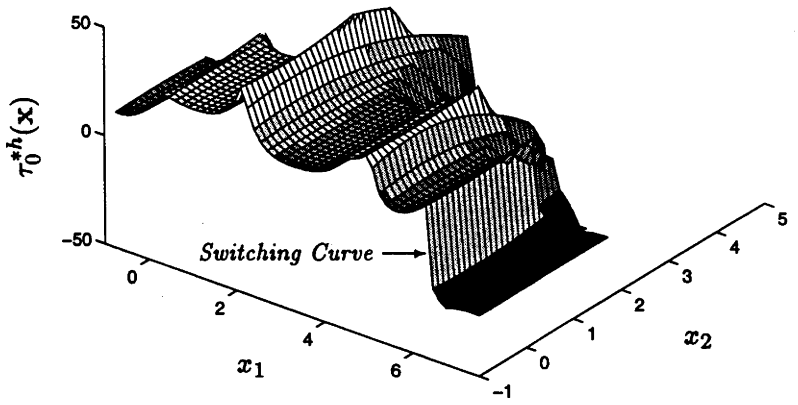


Figure 2.31: Switching Curve in the Optimal Control Policy $\tau_0^{*h}(\mathbf{x})$

2.7 Including a Friction Model

In the examples above, we have assumed no friction terms in the dynamic model (2.18). This is because of the difficulty in adequately modelling such terms. However, if we include a friction model in the dynamics, then we obtain some interesting results.

Consider the new model

$$\hat{\mathbf{M}}(\mathbf{q})\ddot{\mathbf{q}} + \hat{\mathbf{v}}(\mathbf{q}, \dot{\mathbf{q}}) + \hat{\mathbf{d}}(\dot{\mathbf{q}}) = \boldsymbol{\tau} \tag{2.21}$$

where the mass matrix $\hat{\mathbf{M}}(\mathbf{q})$, and the coriolis-centrifugal vector $\hat{\mathbf{v}}(\mathbf{q}, \dot{\mathbf{q}})$, are as before, and where the friction model is described by the discontinuous vector function $\hat{\mathbf{d}}(\dot{\mathbf{q}})$ defined in (A.4) of Appendix A.

The friction model parameters, identified using the same least squares method as before, are shown in Table 2.2.

Parameter	Value	Units	Represents
f_{1p}	15.8909	Nm	Friction terms
f_{1n}	-14.5807	Nm	
f_{2p}	13.1516	Nm	
f_{2n}	-14.0650	Nm	
b_{1p}	44.1334	Nms/rad	
b_{1n}	41.3078	Nms/rad	
b_{2p}	18.8109	Nms/rad	
b_{2n}	16.3550	Nms/rad	

Table 2.2: SCARA Manipulator Friction Parameter Values

The new dynamic model (2.21) is used in the calculation of a new admissible region $\mathcal{A}$, and a new admissible control space $\mathcal{U}_0(\mathbf{x})$. These are shown in Figures 2.32 and 2.33 respectively. Both appear very similar to those for the model which excludes the friction model (Figures 2.4 and 2.5).

However, if we re-formulate and solve the problems of § 2.5.1 and § 2.5.2, then the effect of the friction terms becomes apparent.

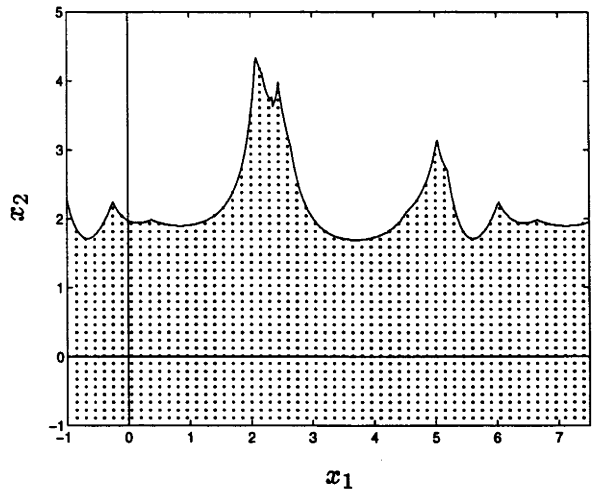


Figure 2.32: Admissible Region - $\mathcal{A}$

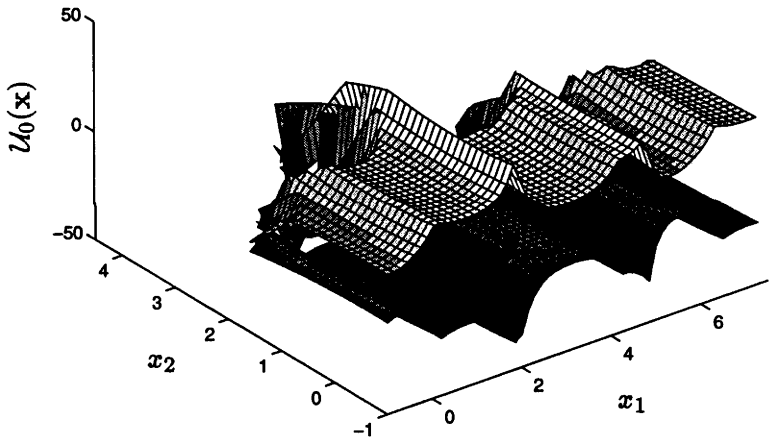


Figure 2.33: Admissible Controls - $\mathcal{U}_0(\mathbf{x})$

In the pure minimum-time case, the time taken to track the path provided by the resulting discounted minimum-time function increases from $t_f \approx 3.746s$ to $t_f \approx 3.870s$. Integration of equations (2.12) using the feedback control policy yields a cycle time of $t_f \approx 3.850s$ (again marginally less than that predicted by the discounted minimum-time function). This increase in time is not wholly unexpected since the friction terms will tend to retard the motion of the manipulator. However, this increase in time provides

great benefit in the accuracy of the tracking. Figure 2.34 reveals that the error has reduced from $\sim O(0.05\ rad)$ to $\sim O(0.01\ rad)$, *c.f.* Figure 2.12.

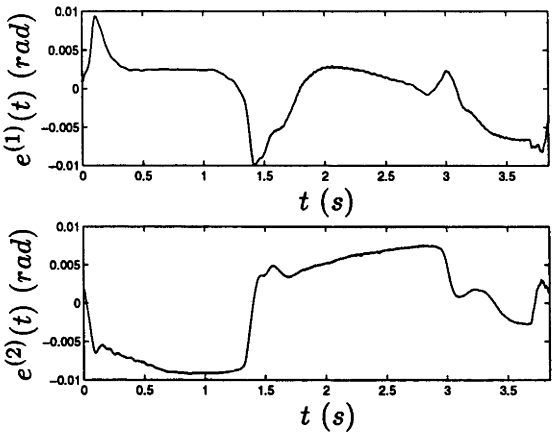


Figure 2.34: End-Effector Tracking Error - Pure Minimum-Time

We obtain similar results for the minimum-time plus quadratic cost example, where the time taken to track the path increases from $t_f \approx 4.303s$ to $t_f \approx 4.400s$, and the error decreases from $\sim O(0.03\ rad)$ to $\sim O(0.01\ rad)$, Figure 2.35.

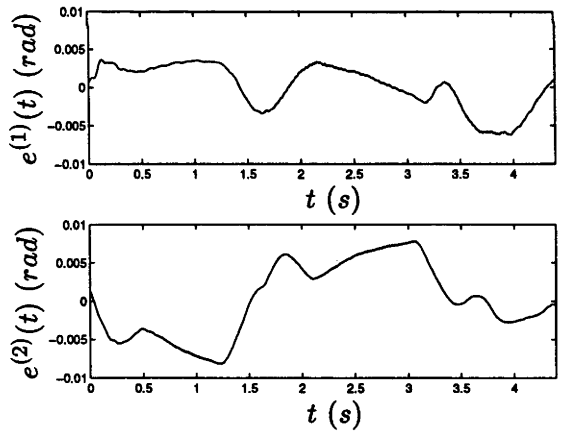


Figure 2.35: End-Effector Tracking Error - Minimum-Time Plus Quadratic Cost

Interestingly, and unlike the earlier examples excluding the friction dynamics, there is only a small advantage in tracking when using the additional quadratic cost. It appears that the error rejection, via the PD part of the computed torque controller, has reached its limits - at least for the gain settings used. Figure 2.36 reveals that the levels of clipping of the commanded torques are significantly reduced compared to the

equivalent problem but without the friction model (Figure 2.23). This suggests that we might be able to increase the controller gains to reduce the tracking errors further.

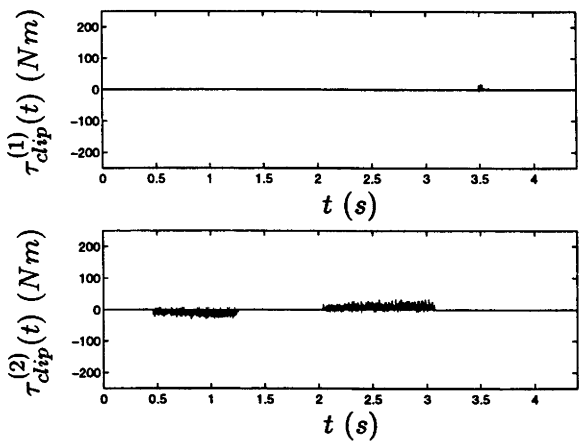


Figure 2.36: Torque Clipping - Minimum-Time Plus Quadratic Cost

Figures 2.37 and 2.38, taken from the pure minimum-time case, show that when the friction terms are included, there is a small increase in the time and number of iterations taken for the numerical scheme to converge (*c.f.* Figures 2.25 and 2.26). More interestingly however, is the fact that the error in the convergence, predicted to be $e(h) \sim O(h^{\frac{1}{2}})$, is sensitive to the inclusion of the friction dynamics, Figures 2.39 and 2.40 (*c.f.* Figures 2.27 and 2.28). We conclude that this is due to the discontinuities in the friction model.

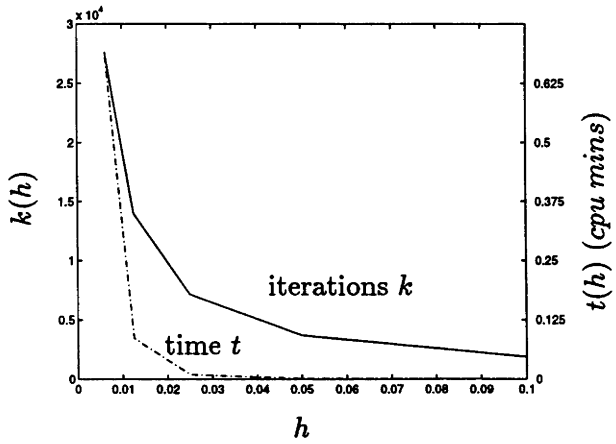


Figure 2.37: Time and iterations to converge versus h

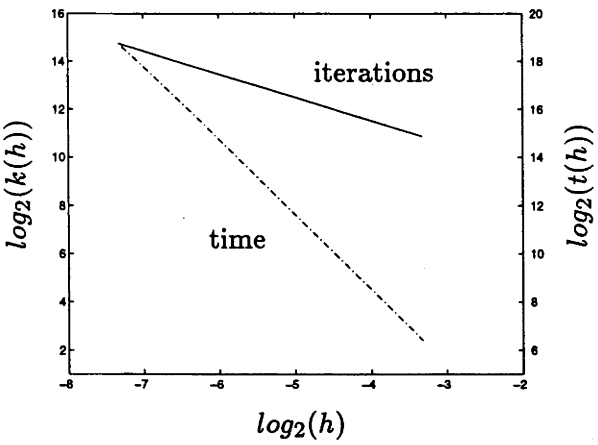


Figure 2.38: log_2(Time and Iterations to Converge) versus $log_2(h)$

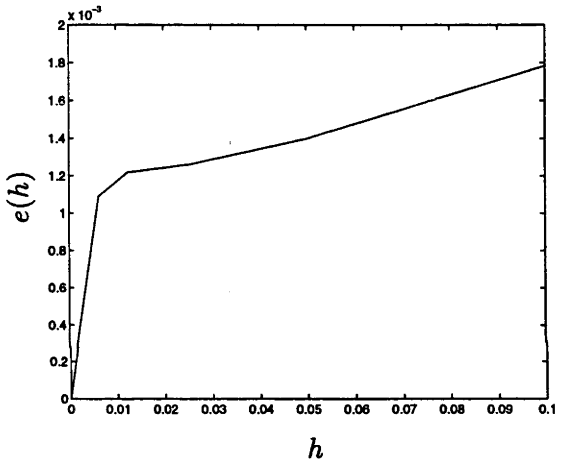


Figure 2.39: Error in $S^h(\cdot)$ versus h

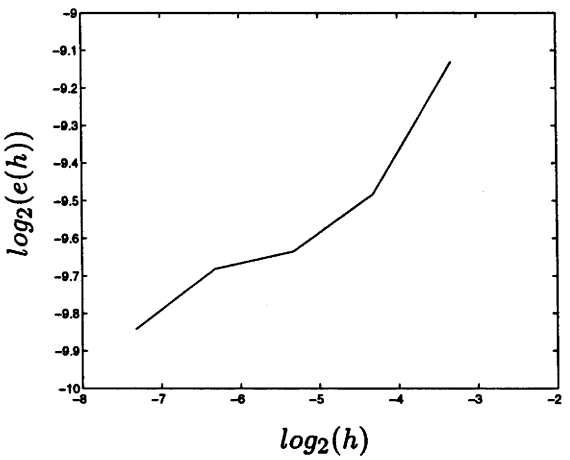


Figure 2.40: log_2(Error in $S^h(\cdot)$) versus $log_2(h)$

2.8 Conclusions

The key findings of our research are:

1. Dynamic programming is computationally feasible for this problem.
2. Solutions of the dynamic programming algorithm can be applied to a real SCARA manipulator with good results.
3. The rate of convergence of the numerical scheme is consistent with that predicted by theory.
4. The minimum-time end-point (target) constraint imposes severe limitations regarding the rate of convergence and computational speed of the dynamic programming algorithm (in particular, commonly used acceleration methods do not work!).
5. The PMP based shooting method is superior to dynamic programming when applied to the standard minimum-time criteria, and is the method of choice.
6. The dynamic programming algorithm can easily be modified to handle more general optimisation criteria, in which case the advantages of the PMP approach diminish (because the optimal control is no longer bang-bang in general, and a two-point boundary value problem needs to be solved).

These issues are expected to manifest themselves in other more involved nonlinear control problems.

Chapter 3

Representing Robot Modelling Errors as Disturbances in Joint Accelerations: Theory and Experiments

Abstract

In this chapter, we propose an approach for identifying robot modelling errors as disturbances in joint accelerations and a theory for relating such disturbances to the performance of robots under computed-torque control. We also present experimental evidence that this theory works well in practice. These results form the basis of our subsequent work in Chapters 4 and 5.

3.1 Introduction

The performance of a robot running under a model-based controller depends on the accuracy of the dynamic model. Thus, to predict the system performance one needs some knowledge of the modelling errors, or disturbances, and some theory for relating those errors to the system performance. Such a theory would be useful, for example, in relating the tuning of a computed-torque controller [21] to the robot's path-tracking performance.

However, the problem of identifying errors in modelling is difficult, in the sense that there is no obvious way to *model* modelling errors. Generally, these errors represent characteristics of the real plant that are too difficult to model, and usually their very structure is unknown.

In this chapter, we investigate the possibility of representing modelling errors as disturbances in joint accelerations. As we show, this representation is attractive for robots under computed-torque control because theory predicts that such disturbances will be related to tracking errors and compensation torques through the linear model of the closed-loop system.

Further, we identify two principle obstacles to applying this theory in practice. The first is that the measurement error may make it impossible to identify the joint acceleration disturbances accurately enough to be useful. The second is that the joint acceleration disturbances must be similar for similar trajectories if the theory is to be used in a predictive sense.

Finally, we show that experimental results taken from our SCARA robot agree very closely with the theory, and thus that the theory is useful in spite of these obstacles. We subsequently use this theory as the basis for our work in Chapters 4 and 5.

3.2 The Theory

Let the robot plant

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{n}(\mathbf{q}, \dot{\mathbf{q}}) = \boldsymbol{\tau}$$

be represented as follows:

$$\hat{\mathbf{M}}(\mathbf{q})\ddot{\mathbf{q}} + \hat{\mathbf{n}}(\mathbf{q}, \dot{\mathbf{q}}) + \hat{\mathbf{M}}(\mathbf{q})\ddot{\mathbf{q}}_d = \boldsymbol{\tau} \,. \tag{3.1}$$

Here $\hat{\mathbf{M}}(\mathbf{q})\ddot{\mathbf{q}} + \hat{\mathbf{n}}(\mathbf{q}, \dot{\mathbf{q}})$ represents the plant model, identified, for example, by least-squares estimation, and $\hat{\mathbf{M}}(\mathbf{q})\ddot{\mathbf{q}}_d$ accounts for any torques that the plant model fails to account for. $\hat{\mathbf{M}}(\mathbf{q})$ is the mass matrix, and $\ddot{\mathbf{q}}_d$ can be interpreted as a *joint acceleration disturbance.*

$\mathbf{q}(t)$, $\dot{\mathbf{q}}(t)$, $\ddot{\mathbf{q}}(t)$ and $\boldsymbol{\tau}(t)$ represent the joint position, velocity, acceleration, and torque trajectories, and $\ddot{\mathbf{q}}_d(t)$ is defined as a function of these trajectories as follows:

$$\ddot{\mathbf{q}}_d \triangleq \hat{\mathbf{M}}^{-1}(\mathbf{q})\{\boldsymbol{\tau} - \hat{\mathbf{M}}(\mathbf{q})\ddot{\mathbf{q}} - \hat{\mathbf{n}}(\mathbf{q}, \dot{\mathbf{q}})\} . \tag{3.2}$$

We note that $\ddot{\mathbf{q}}_d(t)$ can be determined using this equation only if we have knowledge of the input torque trajectory $\boldsymbol{\tau}(t)$. Therefore, we must have knowledge of $\boldsymbol{\tau}(t)$ even if the actuators saturate. In our experimental system, described in Appendix A, the saturation levels are set in software and so we have knowledge of $\boldsymbol{\tau}(t)$ at all times.

The attraction of representing the model errors in terms of $\ddot{\mathbf{q}}_d(t)$ is because if we apply the computed-torque control law

$$\boldsymbol{\tau} = \hat{\mathbf{M}}(\mathbf{q})\{\ddot{\mathbf{q}}_r + \mathbf{k}_v\dot{\mathbf{e}} + \mathbf{k}_p\mathbf{e}\} + \hat{\mathbf{n}}(\mathbf{q}, \dot{\mathbf{q}}) \tag{3.3}$$

to the plant (3.1), *c.f.* Figure 3.1, we obtain the linear system

$$\ddot{\mathbf{e}}(t) + \mathbf{k}_v\dot{\mathbf{e}}(t) + \mathbf{k}_p\mathbf{e}(t) = \ddot{\mathbf{q}}_d(t) . \tag{3.4}$$

Here $\mathbf{e}(t) = \mathbf{q}_r(t) - \mathbf{q}(t)$, where $\mathbf{q}_r(t)$ is the reference trajectory. *Note that equation (3.4) is valid only if saturation does not occur, i.e. if $\boldsymbol{\tau}$ is never greater than the levels of torque available.*

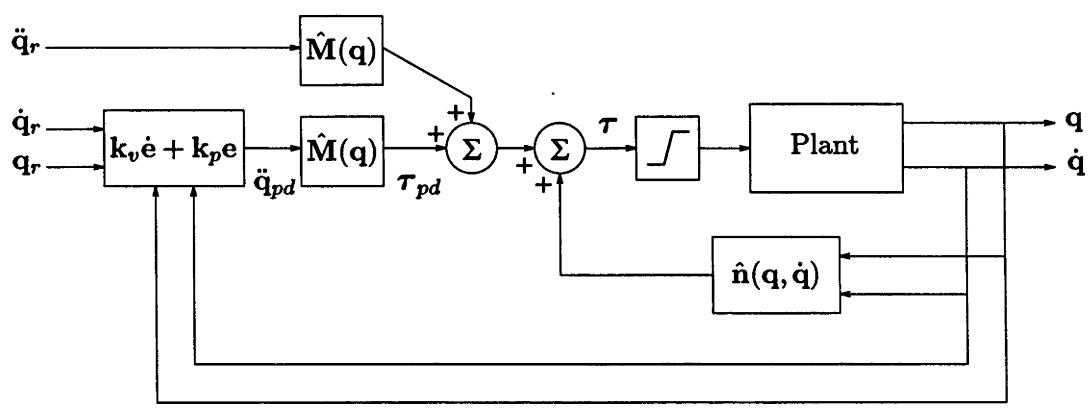


Figure 3.1: The Robot Under Computed Torque Control

This implies that knowledge of $\ddot{\mathbf{q}}_d(t)$, can be used to predict the tracking error $\mathbf{e}(t)$ based on (3.4). Furthermore, knowledge of $\mathbf{e}(t)$ can be used to predict the *compensation*

torques, $\tau_{pd}(t)$, based on the following equation:

$$\tau_{pd}(t) = \hat{\mathbf{M}}(\mathbf{q}(t))\{\mathbf{k}_v\dot{\mathbf{e}}(t) + \mathbf{k}_p\mathbf{e}(t)\} \; . \tag{3.5}$$

Here, $\tau_{pd}(t)$ represents that part of the input torque signal $\tau(t)$ which is generated through the PD-part of the computed-torque control law (3.3) by the error signal $\mathbf{e}(t)$ and its derivative $\dot{\mathbf{e}}(t)$. In turn, this error signal is excited by the disturbance signal $\ddot{\mathbf{q}}_d(t)$ through equation (3.4). Thus, if $\mathbf{e}(t) = 0$ and $\dot{\mathbf{e}}(t) = 0$, then we have $\tau_{pd}(t) = 0$.

Even without exact knowledge of $\ddot{\mathbf{q}}_d(t)$, we can use bounds on it and equations (3.4) and (3.5) to calculate bounds on $\mathbf{e}$ and τ_{pd} as follows:

$$\|\mathbf{e}\|_\infty \leq \frac{1}{\mathbf{k}_p}\|\ddot{\mathbf{q}}_d\|_\infty \; , \tag{3.6}$$

$$|\tau_{pd}(\mathbf{q})| \leq 1.2707 \; |\hat{\mathbf{M}}(\mathbf{q})| \; \|\ddot{\mathbf{q}}_d\|_\infty \; . \tag{3.7}$$

Note that here $\|\cdot\|_\infty$ represents the vector l_∞ norm, and that these bounds are derived from (3.4) and (3.5) using linear system theory and assuming critical damping ($\mathbf{k}_v = 2\sqrt{\mathbf{k}_p}$), see Appendix 3.A.

3.3 The Experiments

It is trivial to generate computer simulations that support the theory in § 3.2. With *a priori* knowledge of $\ddot{\mathbf{q}}_d(t)$, one can integrate (3.4) to compute $\mathbf{e}(t)$ exactly, and then compute $\tau_{pd}(t)$ based on (3.5). One can also estimate $\|\ddot{\mathbf{q}}_d\|_\infty$ from one or more simulations, then use it to predict bounds (3.6) and (3.7) for new simulations. As long as the impact of modelling errors is similar, results will confirm theory.

However, it is quite another matter to show that the theory can be applied in practice. The biggest obstacle is that clean measurements are not available. In particular, it is usually necessary to numerically estimate the signals $\dot{\mathbf{q}}(t)$ and $\ddot{\mathbf{q}}(t)$ from encoder measurements of $\mathbf{q}(t)$, and such estimation introduces high levels of noise into the acceleration disturbance signal $\ddot{\mathbf{q}}_d(t)$. There is, however, a factor which counteracts this noise; that when integrating $\ddot{\mathbf{q}}_d(t)$ to obtain the error, equation (3.4) will act as a low pass filter.

Prior to assessing how well the theory works in practice, let us first present the robot system on which we have performed our experiments.

3.3.1 Experimental System

Our experimental manipulator is the 4 degree-of-freedom SCARA arm described in Appendix A. For simplicity, we restrict the motion to be planar, driven by only the first and second joint motors.

The planar dynamics of the SCARA are modelled as

$$\hat{\mathbf{M}}(\mathbf{q})\ddot{\mathbf{q}} + \hat{\mathbf{v}}(\mathbf{q}, \dot{\mathbf{q}}) + \hat{\mathbf{d}}(\dot{\mathbf{q}}) = \boldsymbol{\tau}$$

where $\hat{\mathbf{M}}(\mathbf{q})$, $\hat{\mathbf{v}}(\mathbf{q}, \dot{\mathbf{q}})$ and $\hat{\mathbf{d}}(\dot{\mathbf{q}})$ represent the mass matrix, coriolis-centrifugal and friction vectors respectively, defined in equations (A.2), (A.3) and (A.4).

The parameter values for $\hat{\mathbf{M}}(\mathbf{q})$, $\hat{\mathbf{v}}(\mathbf{q}, \dot{\mathbf{q}})$ and $\hat{\mathbf{d}}(\dot{\mathbf{q}})$, identified using the standard least squares technique, are shown in Table 3.1.

The torque limits for this robot are given by

$$\left.\begin{array}{rcll} \underline{\tau}(\dot{\mathbf{q}}) & = & \max & \{-309.42, -1146.0 - 984.987\dot{\mathbf{q}}\} \\ \overline{\tau}(\dot{\mathbf{q}}) & = & \min & \{+309.42, +1146.0 - 984.987\dot{\mathbf{q}}\} \end{array}\right\}, \tag{3.8}$$

and are due to current and voltage limits of $6A$ and $40V$ respectively.

3.3.2 Experiments and Results

Figures 3.2, 3.3 and 3.4 show the results from an experiment intended to assess how well the theory works in practice.

A computed-torque controller was applied to the first two joints of the SCARA. The chosen reference trajectory traces a straight line in joint space and was slow enough to avoid saturation of either actuator.

The experiment was performed and $\dot{\mathbf{q}}(t)$ and $\ddot{\mathbf{q}}(t)$ were estimated from the measured output $\mathbf{q}(t)$ by finite difference. The acceleration disturbance $\ddot{\mathbf{q}}_d(t)$ was generated using equation (3.2) and is displayed in Figure 3.2. Note the high levels of noise present. A

Parameter	Value	Units	Represents
f_{11}	0.3974	Nms^2/rad	Inertia terms
$f_{12} = f_{21}$	0.1987	Nms^2/rad	
g_{11}	3.7694	Nms^2/rad	
$g_{12} = g_{21}$	1.8847	Nms^2/rad	
h_{11}	37.9687	Nms^2/rad	
$h_{12} = h_{21}$	2.9851	Nms^2/rad	
h_{22}	17.2015	Nms^2/rad	
v_{1s}	-3.7694	Nms^2/rad^2	Centrifugal
v_{2s}	1.8847	Nms^2/rad^2	and coriolis
w_{1s}	-1.8847	Nms^2/rad^2	terms
v_{1c}	0.3974	Nms^2/rad^2	
v_{2c}	-0.1987	Nms^2/rad^2	
w_{1c}	0.1987	Nms^2/rad^2	
f_{1p}	15.8909	Nm	Friction terms
f_{1n}	-14.5807	Nm	
f_{2p}	13.1516	Nm	
f_{2n}	-14.0650	Nm	
b_{1p}	44.1334	Nms/rad	
b_{1n}	41.3078	Nms/rad	
b_{2p}	18.8109	Nms/rad	
b_{2n}	16.3550	Nms/rad	
l_1	0.6100	m	Length of links
l_2	0.5800	m	1 and 2

Table 3.1: SCARA Manipulator Parameter Data

prediction of the error, $\hat{\mathbf{e}}(t)$, was generated by integrating $\ddot{\mathbf{q}}_d(t)$ through equation (3.4) with results shown in Figure 3.3. The compensation torque, $\hat{\boldsymbol{\tau}}_{pd}(t)$, shown in Figure 3.4, was predicted based on equation (3.5) using $\hat{\mathbf{e}}(t)$, $\dot{\hat{\mathbf{e}}}(t)$ and $\mathbf{q}_r(t)$ in place of $\mathbf{e}(t)$, $\dot{\mathbf{e}}(t)$ and $\mathbf{q}(t)$, respectively.

The estimate of the error, $\hat{\mathbf{e}}(t)$, follows the actual error $\mathbf{e}(t)$ very closely. Similarly, the estimate $\hat{\boldsymbol{\tau}}_{pd}(t)$ of $\boldsymbol{\tau}_{pd}(t)$ is very accurate. This indicates that measurement noise is not an overwhelming problem, at least for our robot which is equipped with high resolution encoders (360,000 counts per revolution). But can the theory be applied in a predictive sense ?

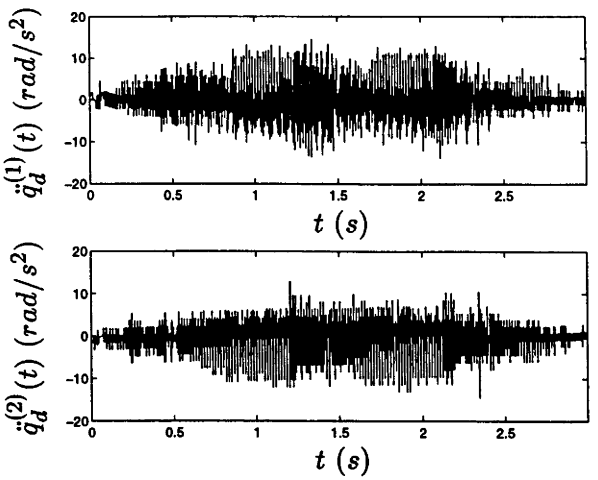


Figure 3.2: Acceleration Disturbance $\ddot{\mathbf{q}}_d(t)$

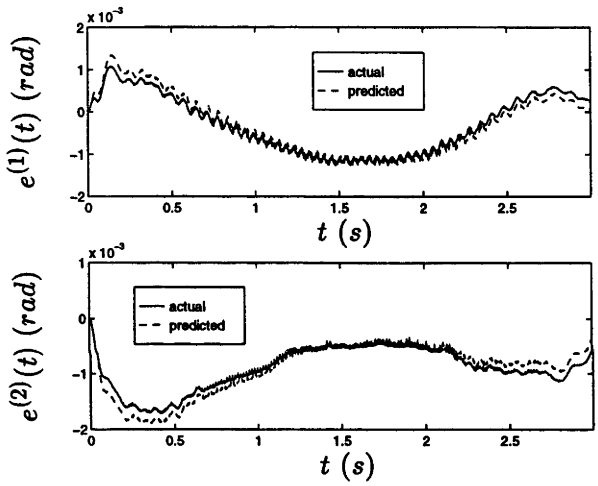


Figure 3.3: Actual and Estimated Errors

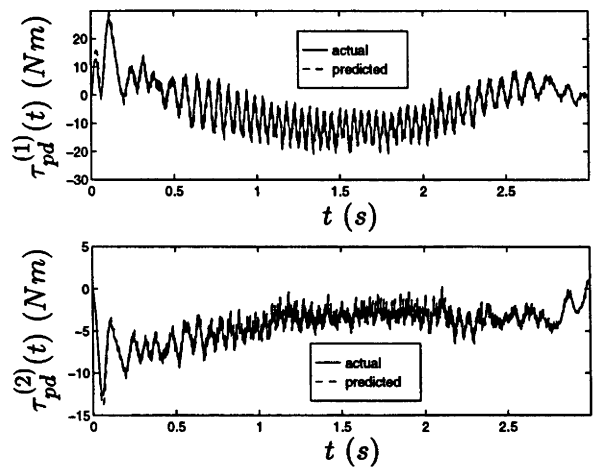


Figure 3.4: Actual and Estimated Compensation Torques

Figures 3.5 to 3.7 show results from 6 experiments which attempt to assess whether the equations (3.4) and (3.5) and the inequalities (3.6) and (3.7), together with knowledge of $\ddot{\mathbf{q}}_d$ can be used to relate the tuning of the controller to the tracking performance. The robot system described above was driven with the same reference trajectory as before, but using 6 different values of $\mathbf{k}_p$ (and $\mathbf{k}_v = s\sqrt{\mathbf{k}_p}$). The data measured from the experiments is shown as asterisks.

Two methods for predicting $\|\mathbf{e}\|_\infty$ and $\|\boldsymbol{\tau}_{pd}\|_\infty$ were considered. The first, based on the inequalities (3.6) and (3.7), is represented by the solid lines in Figures 3.6 and 3.7. These assume $\|\ddot{\mathbf{q}}_d\|_\infty = [0.93, 0.87]^T$ (rad/s^2), Figure 3.5. Note that when calculating $\|\ddot{\mathbf{q}}_d\|_\infty$, we first applied a $10Hz$ low pass filter to $\ddot{\mathbf{q}}_d(t)$ to remove the effects of noise (if this is not done, then the estimate of $\|\mathbf{q}_d\|_\infty$ will be greatly exaggerated, and our estimates of the bounds on $\mathbf{e}$ and $\boldsymbol{\tau}_{pd}$ will be very much greater than the real bounds). Based on these results, we concluded that this first method works, but that it is too conservative.

The second method, represented by circles in Figures 3.6 and 3.7, is to predict bounds on $\mathbf{e}$ by integrating $\ddot{\mathbf{q}}_d(t)$ through equation (3.4), and then to apply the resulting $\hat{\mathbf{e}}(t)$ and $\dot{\hat{\mathbf{e}}}(t)$ in equation (3.5) to predict $\boldsymbol{\tau}_{pd}$ (in this case $\ddot{\mathbf{q}}_d(t)$ is not filtered since (3.4) acts as a low pass filter). The actual data is consistent with theory. Indeed, the data calculated using equations (3.4) and (3.5) provides an excellent fit for both the tracking errors and compensation torques.

3.4 Conclusion

We have proposed a method for representing modelling errors as disturbances in joint acceleration, and a theory for relating such disturbances to the tracking errors and compensation torques for robots under computed-torque control. Experimental results confirm that the theory works very well for our robotic system. In the next chapter, we will apply this knowledge to the problem of planning robust time-optimal trajectories to meet a prescribed tracking tolerance.

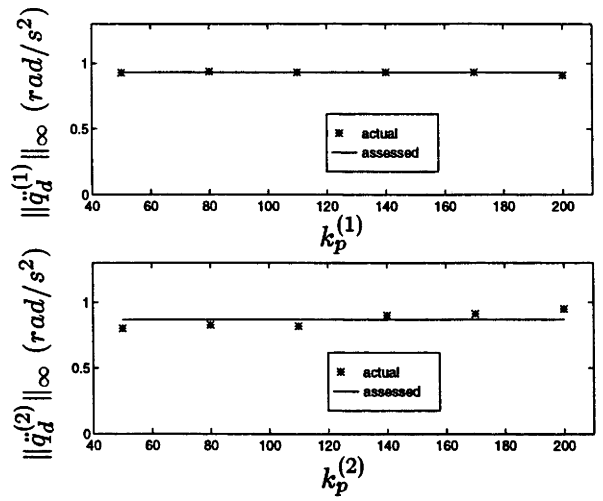


Figure 3.5: Maximum Joint Acceleration Disturbances

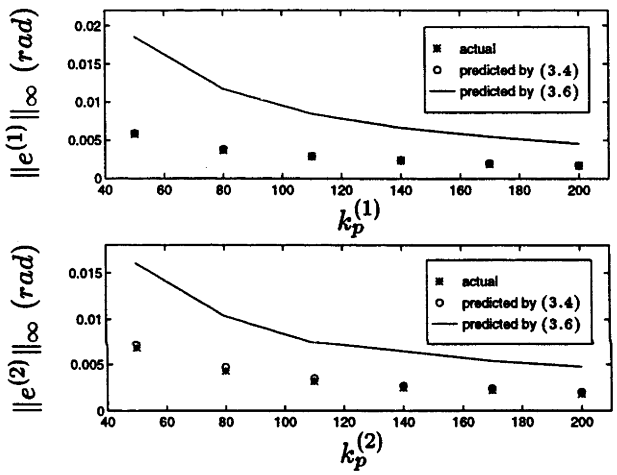


Figure 3.6: Maximum Errors - Actual and Predicted

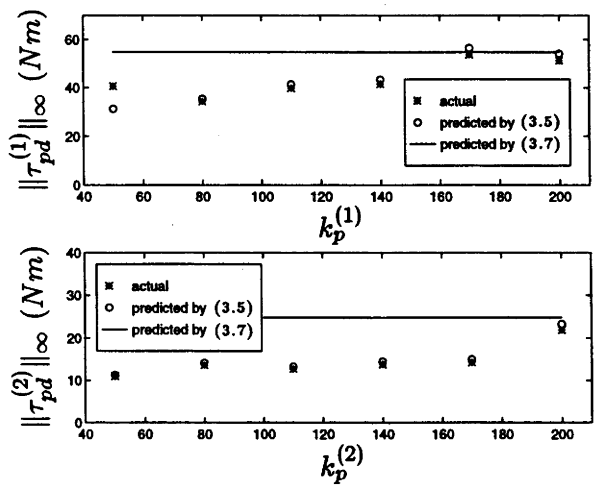


Figure 3.7: Maximum Compensation Torques - Actual and Predicted

Appendix

3.A Calculation of the Error and Compensation Torque Bounds

To choose controller gains, we assume critical damping, i.e. $\mathbf{k}_v = 2\sqrt{\mathbf{k}_p}$. Then the steady state error in the closed loop system (3.4),

$$\ddot{\mathbf{e}}(t) + \mathbf{k}_v\dot{\mathbf{e}}(t) + \mathbf{k}_p\mathbf{e}(t) = \ddot{\mathbf{q}}_d(t)\,, \tag{3.9}$$

in response to a step input is equal to $1/\mathbf{k}_p$.

Thus, assuming no saturation, the tracking error is limited by

$$\|\mathbf{e}\|_\infty \leq \frac{1}{\mathbf{k}_p}\|\ddot{\mathbf{q}}_d\|_\infty$$

which means that we can satisfy $\|\mathbf{e}\|_\infty \leq \epsilon$ by choosing $\mathbf{k}_p$ such that

$$\mathbf{k}_p \geq \frac{\|\ddot{\mathbf{q}}_d\|_\infty}{\epsilon}\,.$$

Here, $\|\cdot\|_\infty$ represents the vector l_∞ norm, i.e. and e.g.

$$\|\ddot{\mathbf{q}}_d\|_\infty \triangleq \begin{bmatrix} \|\ddot{q}_d^{(1)}\|_\infty \\ \cdot \\ \cdot \\ \|\ddot{q}_d^{(n)}\|_\infty \end{bmatrix} = \begin{bmatrix} \max_t |\ddot{q}_d^{(1)}(t)| \\ \cdot \\ \cdot \\ \max_t |\ddot{q}_d^{(n)}(t)| \end{bmatrix}\,.$$

We can then compute a compensation torque, or backoff, via equation (3.5)

$$\tau_{pd}(t) = \hat{\mathbf{M}}(\mathbf{q}(t))\{\mathbf{k}_v\dot{\mathbf{e}}(t) + \mathbf{k}_p\mathbf{e}(t)\} \triangleq \hat{\mathbf{M}}(\mathbf{q}(t))\ddot{\mathbf{q}}_{pd}(t) \tag{3.10}$$

using the Hölder inequality

$$\|\ddot{\mathbf{q}}_{pd}\|_\infty \leq \|\mathbf{g}\|_1\|\ddot{\mathbf{q}}_d\|_\infty\,. \tag{3.11}$$

Here, $\mathbf{g}(t)$ is the impulse response of the transfer function $\mathbf{G}(s)$ from the disturbance $\ddot{\mathbf{Q}}_d(s)$ to the PD errors $\ddot{\mathbf{Q}}_{pd}(s)$, which can be derived from equations (3.9) and (3.10). Assuming critical damping, $\mathbf{G}(s)$ is given by

$$\mathbf{G}(s) = \frac{2\sqrt{\mathbf{k}_p}s + \mathbf{k}_p}{s^2 + 2\sqrt{\mathbf{k}_p}s + \mathbf{k}_p} = \frac{\sqrt{\mathbf{k}_p}}{s + \sqrt{\mathbf{k}_p}} + \frac{\sqrt{\mathbf{k}_p}s}{(s + \sqrt{\mathbf{k}_p})^2}\,,$$

and $\mathbf{g}(t)$ is

$$\mathbf{g}(t) = \mathcal{L}^{-1}\{\mathbf{G}(s)\} = te^{-\sqrt{\mathbf{k}_p}t} + e^{-\sqrt{\mathbf{k}_p}t} ,$$

whence

$$\|\mathbf{g}\|_1 = \int_0^\infty |te^{-\sqrt{\mathbf{k}_p}t} + e^{-\sqrt{\mathbf{k}_p}t}|dt = 1.2707 . \tag{3.12}$$

Note that the integral in (3.12) is one sided since each $\ddot{\mathbf{q}}_d^{(i)}(t)$ is defined only for $t \geq 0$.

Based on equations (3.10), (3.11) and (3.12), we define the bound on the compensation torque as a function of the configuration via

$$|\boldsymbol{\tau}_{pd}(\mathbf{q})| \leq 1.2707\, |\hat{\mathbf{M}}(\mathbf{q})|\, \|\ddot{\mathbf{q}}_d\|_\infty . \tag{3.13}$$

Note that this is independent of the controller gains $\mathbf{k}_p$. This suggests that the compensation torque defined in this way can be considered an upper bound for the torques that will be generated through the PD loop during an experiment.

Chapter 4

Robust Time-Optimal Trajectory Planning for Robot Manipulators

Abstract

In this chapter, we propose a solution to the problem of determining a robust time-optimal path tracking control strategy to meet a prescribed tolerance in the presence of plant uncertainty. Our approach makes use of the theory in Chapter 3 which relates joint acceleration disturbances to tracking errors and compensation torques. This theory allows us to calculate the torque levels that need to be held in reserve during trajectory planning and the controller gains that are required to reject the expected levels of disturbance. Our experiments show that if these levels of torque are held in reserve and if the controllers are tuned in this way, then the robot performs near time-optimal tracking meeting the specified tolerance for tracking accuracy.

4.1 Introduction

Well known approaches to the problem of planning time-optimal trajectories when motion is constrained to a path utilise the path constraint to reduce the dimension of the problem. The trajectories are then planned using shooting methods or by dynamic

programming [8, 10, 51, 58, 60, 61, 66].

The trajectories resulting from such methods are theoretically time-optimal, but are not suitable for practical application because they fail to address some important issues. There are two major obstacles. The first is that the methods assume exact knowledge of the robot dynamics. The second is that disturbance rejection requires the implementation of a tracking controller whose dynamics were not considered during the planning of the trajectories. Both of these invalidate the planned trajectories and can cause a loss of control when actuators become over-saturated. One way around this problem is to plan trajectories using conservative torque bounds and then hope that the torques held in reserve will be sufficient to cope with the un-modelled dynamics. However, without some way of determining the actual levels of torque that need to be held in reserve, this approach is arbitrary, ensuring neither optimality nor tracking performance.

Some research has addressed the issue of coping with disturbances. There are two approaches. The first involves quantifying model parameter uncertainties and then making allowances for them when planning the trajectory [34, 50, 62, 68]. The second involves modifying the trajectories on-line in order to avoid the loss of control due to disturbances [2, 3, 22, 23, 24]. (See the discussion in Chapter 1)

However, all of these approaches have some element of *conservativeness* built into them. Further, none of them consider a fundamental question: *Given a specified tolerance for tracking by the end-effector, how do we ensure that it will be achieved*?

In this chapter, we propose the following new approach to robust time-optimal trajectory planning:

1. We identify the disturbances as accelerations rather than as torques, using the theory developed in Chapter 3. This theory shows that acceleration disturbances can be linked to the tracking error. This link is through the error dynamics of the plant under a PD computed-torque controller.

2. This link allows the PD gains to be set to reject expected levels of disturbance to meet the tracking tolerance specification.

3. Once the gains have been selected, we can calculate the torque that needs to be held in reserve during trajectory planning.

In § 4.2 we discuss how we apply the theory presented in Chapter 3. In § 4.3 we present our experimental system and in § 4.4 we present some performance measures for the experiments. In § 4.5 we present some experimental results which illustrate the method. In § 4.6 we demonstrate the robustness of the method by applying it to several randomly chosen paths. In running the experiments, our experimental manipulator developed backlash. After fixing this problem, we re-ran some of the experiments. The results of these are presented in § 4.7. Finally, in § 4.8 we draw conclusions.

4.2 Application of the Theory

In this chapter, we wish to apply the theory presented in Chapter 3 in a predictive sense. Our purpose is to predict the errors which will be excited by the disturbances during the time-optimal control of a manipulator, and to identify the torque levels and controller gains required to reject these errors to a specified level.

We have considered two approaches to using the theory. The first, § 4.2.1, considers identifying the worst case acceleration disturbance across all configurations, joint velocities and accelerations, then planning controls and trajectories that can reject such disturbances at any time. (In practice we would use a suitably demanding subset of this region). Whilst this would lead to a globally robust result, one where control strategies could be defined for any path without recourse to re-identifying the acceleration disturbances, this approach may be too conservative in practice (it is in our experimental system). This is because the worst case compensation torques may be larger than the torques that are available. Thus, we have explored a second approach, § 4.2.2, which identifies the acceleration disturbance in the region of the desired trajectory.

4.2.1 A Conservative Global Solution

Let us define a worst case acceleration disturbance $\|\ddot{\mathbf{q}}_d\|_\infty$ from n experiments $\ddot{\mathbf{q}}_d^{(i)}(t)$ as follows:

$$\|\ddot{\mathbf{q}}_d\|_\infty \triangleq \max_{all\ experiments\ i} \|\ddot{\mathbf{q}}_d^{(i)}\|_\infty$$

where

$$\|\ddot{\mathbf{q}}_d^{(i)}\|_\infty = \max_t \{|\ddot{\mathbf{q}}_d^{(i)}(t)|\} \ .$$

Here, $\|\ddot{\mathbf{q}}_d^{(i)}\|_\infty \triangleq \max_t |\ddot{\mathbf{q}}_d^{(i)}(t)|$ represents the vector l_∞ norm of the acceleration disturbance vector $\ddot{\mathbf{q}}_d^{(i)}(t)$.

We now assume that such disturbance levels can occur at any time and we wish to ensure that a given path is tracked to a prescribed tolerance. In practice, one might prescribe the tracking tolerance as a restriction in Cartesian space, for example $\pm 2mm$. However, it is possible to convert this to an equivalent restriction in the joint space through the inverse kinematics. For simplicity, we assume that we are given a joint space restriction, i.e. that $\|\mathbf{e}\|_\infty \leq \epsilon$.

In Appendix 3.A of Chapter 3 we show that, assuming no saturation and critical damping, $\|\mathbf{e}\|_\infty \leq \epsilon$ can be satisfied by choosing the controller gains $\mathbf{k}_p$ such that

$$\mathbf{k}_p \geq \frac{\|\ddot{\mathbf{q}}_d\|_\infty}{\epsilon} \ ,$$

and that to ensure that saturation does not occur, the compensation torque, or backoff,

$$|\boldsymbol{\tau}_{pd}(\mathbf{q})| = 1.2707\ |\hat{\mathbf{M}}(\mathbf{q})|\ \|\ddot{\mathbf{q}}_d\|_\infty \tag{4.1}$$

is required. Note that this is independent of $\mathbf{k}_p$. This suggests that the backoff defined in this way can be considered a worst case backoff, and will produce an extremely conservative "time-optimal" trajectory.

As discussed in the introduction above, if $\|\ddot{\mathbf{q}}_d\|_\infty$ is too large, then the compensation torques calculated through (4.1) may be greater than those available at some or all configurations. In this case, this approach would not be implementable. Even if we were to relax the identification of the disturbance globally and restrict ourselves to

identifying the disturbance only in the region of the desired trajectory, the compensation torques calculated using this method might be greater than those available at some configurations. We have found that this is the case for our own robot system.

Thus, we present the following alternative approach, which identifies the disturbance in the region of the desired trajectory, and then uses the following approach to identify a less conservative, yet still robust solution.

4.2.2 A Less Conservative Local Solution

Our strategy for obtaining a less conservative solution is to identify $\ddot{\mathbf{q}}_d(t)$ for a trajectory close to the robust "time-optimal" trajectory we seek, and then to use the results to choose the controller gains, compensation torques, and a new reference trajectory for a second experiment. The validity of this approach rests upon our assumption that the disturbance signal $\ddot{\mathbf{q}}_d(t)$ will be approximately the same for similar trajectories. This is the basis for the identification procedure described below. Note that, although the theory for predicting tracking errors, equation (3.4), is only valid if saturation does not occur, the disturbance acceleration $\ddot{\mathbf{q}}_d(t)$ can always be computed, using equation (3.2), whether or not saturation occurs.

The Identification Procedure

- We assume that we are given the path in the joint space, parameterised as a function of the path parameter s viz $\mathbf{q}_r = \mathbf{f}(s)$, the nominal robot dynamics, torque limits of the form

$$\underline{\tau}_i(\mathbf{q}, \dot{\mathbf{q}}) \leq \tau_i \leq \overline{\tau}_i(\mathbf{q}, \dot{\mathbf{q}}) \,, \tag{4.2}$$

  and the error tolerance ϵ.

1. Calculate the "theoretical" time-optimal joint trajectory $\mathbf{q}_r(t)$ based on the model of the robot plant and using the torque limits (4.2) - using, for example, the

shooting method of Bobrow *et al.* [8], or the dynamic programming method described in Chapter 2.

2. Choose reasonable values for the experimental system gains $\mathbf{k}_p$. (We choose critical damping, i.e. $\mathbf{k}_v = 2\sqrt{\mathbf{k}_p}$.)

3. Drive the experimental system using the time-optimal joint trajectory $\mathbf{q}_r(t)$ as reference.

4. Calculate $\ddot{\mathbf{q}}_d(t)$ based on the output of this experiment using equation (3.2).

5. Search for the gains $\mathbf{k}_p$ which produce $\|\hat{\mathbf{e}}\|_\infty = \epsilon$ when integrating $\ddot{\mathbf{q}}_d(t)$ through the error system

$$\ddot{\hat{\mathbf{e}}}(t) + \mathbf{k}_v\dot{\hat{\mathbf{e}}}(t) + \mathbf{k}_p\hat{\mathbf{e}}(t) = \ddot{\mathbf{q}}_d(t)\ , \tag{4.3}$$

and then calculate the associated compensation torques as a function of the path position, via

$$\boldsymbol{\tau}_{pd}(s) = \hat{\mathbf{M}}(\mathbf{q}_r(s))\left\{\mathbf{k}_v\dot{\hat{\mathbf{e}}}(s) + \mathbf{k}_p\hat{\mathbf{e}}(s)\right\}\ . \tag{4.4}$$

6. Place an envelope around $\boldsymbol{\tau}_{pd}(s)$ and mirror in the zero-axis to yield $\underline{\boldsymbol{\tau}}_{pd}(s)$ and $\overline{\boldsymbol{\tau}}_{pd}(s)$ such that $\underline{\boldsymbol{\tau}}_{pd}(s) = -\overline{\boldsymbol{\tau}}_{pd}(s)$, *c.f.* Figure 4.6. We do this because of the potential for oscillation in $\boldsymbol{\tau}_{pd}(s)$, often present in $\dot{\hat{\mathbf{e}}}(t)$.

7. Calculate a new time-optimal trajectory $\mathbf{q}_r(t)$ with $\underline{\boldsymbol{\tau}}_{pd}(s)$ and $\overline{\boldsymbol{\tau}}_{pd}(s)$ held in reserve, i.e.

$$\underline{\boldsymbol{\tau}}(s,\dot{s}) - \underline{\boldsymbol{\tau}}_{pd}(s) \le \boldsymbol{\tau} \le \overline{\boldsymbol{\tau}}(s,\dot{s}) - \overline{\boldsymbol{\tau}}_{pd}(s)\ , \tag{4.5}$$

where $\underline{\boldsymbol{\tau}}(s,\dot{s})$ and $\overline{\boldsymbol{\tau}}(s,\dot{s})$ are defined by (4.2) with $\mathbf{q} = \mathbf{f}(s)$ and $\dot{\mathbf{q}} = \mathbf{f}'(s)\dot{s}$.

8. Set the experimental system gains $\mathbf{k}_p$ to be those which yielded $\|\hat{\mathbf{e}}\|_\infty = \epsilon$ in 5.

9. Drive the experimental system using the time-optimal joint trajectory $\mathbf{q}_r(t)$ as reference.

4.3 Experimental System

The method will be implemented on the first two links of our experimental manipulator, the 4 degree-of-freedom SCARA arm described in Appendix A. The model parameters and torque limits are those defined in Table 3.1 and equation (3.8) respectively in the previous chapter.

There are two comments on technical details to be made here. The first is that the experimental system we used displays some mechanical resonance at approximately $9 - 10Hz$. The implication of this is that there is a limitation on the gains that can be applied, particularly to the first joint, otherwise the closed-loop system exhibits unstable behaviour.

The second concerns our ability to feed back to the controller the raw joint velocity data, calculated numerically, without first filtering it. Our earlier experiments show that because the resolution of the encoders is high (360,000 counts per revolution), the magnitude of the noise introduced by the numerical differentiation process is relatively small and so we do not need to filter $\dot{\mathbf{q}}$ in the feedback loop of the controller. If we did need to do so, then our theory would have to be modified to take account of this. Further, the more prominent noise introduced when taking second derivatives, $\ddot{\mathbf{q}}(t)$, for use in the identification procedure poses no problem since the error system (3.4) through which we integrate the resulting disturbance signal $\ddot{\mathbf{q}}_d(t)$ acts as a low pass filter.

4.4 Performance Measures

Performance will be assessed based on three measures; the tracking error compared to the prescribed tracking tolerance, the torque utilisation τ_{util}, and the torque command clipping τ_{clip}.

The torque utilisation, $\tau_{util}(t)$, defined in (4.6), is a number in the range 0 to 1 which indicates how completely the range of available torques is utilised. It will be 0 if $\boldsymbol{\tau}$ is in

the centre of the hypercube of available torques, and 1 if any τ_i is on a boundary. The bang-bang nature of time-optimal solutions suggest that it is desirable to have $\tau_{util}(t)$ close to or equal to 1. However, $\tau_{util}(t) = 1$ is not desirable if it results from the clipping of commanded torques. This is because clipping invalidates the error model (3.4) on which the developed theory is founded. For this reason, we define $\tau_{clip}(t)$, (4.7), as a measure of how much, if any, of the commanded torque is clipped. $\tau_{clip}(t) = 0$ implies no clipping, while $\tau_{clip}(t) > 0$ is a normalised measure of the most severely clipped joint torque. The sum $\tau_{util}(t) + \tau_{clip}(t)$ can be interpreted as the maximum normalised joint torque command, where normalisation is with respect to the half-range of available torques of each joint.

Finally, we define the quantities τ_{util} and τ_{clip}, (4.8), to be measures of the utilisation and clipping averaged over the whole experiment.

$$\tau_i^{norm}(t) \triangleq \frac{|\tau_i(t) - \tau_i^{ave}(t)|}{(\underline{\tau}_i(t) - \overline{\tau}_i(t))/2} \quad : \quad \tau_i^{ave}(t) = \frac{\overline{\tau}_i(t) + \underline{\tau}_i(t)}{2},$$

$$\tau_{util}(t) = \min\left\{1, \max_i [\tau_i^{norm}(t)]\right\}, \tag{4.6}$$

$$\tau_{clip}(t) = \max\left\{0, \max_i [\tau_i^{norm}(t)] - 1\right\}, \tag{4.7}$$

$$\tau_{util} = \frac{\sum_t \tau_{util}(t)}{t}, \quad \tau_{clip} = \frac{\sum_t \tau_{clip}(t)}{t}. \tag{4.8}$$

We note that the torque utilisation measure is defined similarly to that provided by Dahl [23]. However, Dahl does not provide a measure to indicate clipping of the commanded torques.

4.5 Example

In this section, we present some experimental results that illustrate application of the method to one path tracking task. In the next section we present results from ten randomly chosen path tracking tasks by which we wish to demonstrate the robustness of the method.

The task was to track the following joint space path in minimum-time, and with joint errors less than $\epsilon = (0.005, 0.0025)^T$ *rad*, which equates to 3mm-4mm in the end-effector position:

$$\mathbf{q}(s) = \begin{pmatrix} q_1(s) \\ q_2(s) \end{pmatrix} = \begin{pmatrix} \frac{\pi}{2}\sin(s) \\ \frac{\pi}{2}\cos(\frac{3}{2}s) \end{pmatrix}, \quad s \in [0, 2\pi] .$$

The desired path is shown in Cartesian space, Figure 4.1.

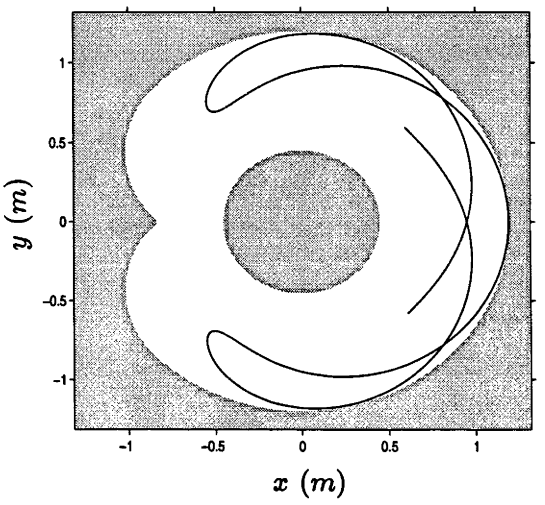


Figure 4.1: Cartesian Space Path

Note that all trajectories were calculated using the trajectory planning method described by Bobrow *et al.* in [8]. Note also that we set the initial values of the controller gains $\mathbf{k}_p$ to be $\mathbf{k}_p = (100, 100)^T$ (which equates to a natural frequency of 10 rad/s, a figure commonly used in literature).

Our initial "theoretically" time-optimal trajectory was planned based on the model dynamics and assuming the actual torque bounds. This is shown in the phase plane plot in Figure 4.2. Application of the joint trajectory which corresponds to this path trajectory as the reference input to the closed-loop system yielded the errors shown in Figure 4.3. Clearly, the tracking is well outside the prescribed tolerance, which is not surprising since the plot of the normalised command torque reveals that one or more of the commanded torques is being clipped for a large proportion of the time, Figure 4.4.

(The maximum errors are $|\mathbf{e}| \approx (0.0117, 0.0139)^T$ *rad.* This compares favourably with the "best" results that we could obtain when driving the system with the same joint trajectory but using standard joint PD controllers, $|\mathbf{e}| \approx (0.0244, 0.0151)^T$ *rad.*)

The acceleration disturbance $\ddot{\mathbf{q}}_d(t)$, calculated from the output of this initial closed-loop experiment, is displayed as a function of the path position $s \equiv s(t)$ in Figure 4.5. Note that the disturbance signals are much higher at certain points along the path than at others.

We then integrated $\ddot{\mathbf{q}}_d(t)$ through the error system (4.3) with differing gains until the estimate of the errors $\hat{\mathbf{e}}(t)$, shown as the solid lines in Figure 4.7, satisfied the prescribed tolerance. The resulting gains which achieved this were $\mathbf{k}_p = (191.4, 942.2)^T$.

The compensation torque $\boldsymbol{\tau}_{pd}(s)$ was then estimated as a function of the path position s using equation (4.4), and the bounding functions $\underline{\boldsymbol{\tau}}_{pd}(s)$ and $\overline{\boldsymbol{\tau}}_{pd}(s)$ determined as shown in Figure 4.6. Note that, like the disturbance signals, these are much lower at certain points along the path than at others (in fact they are greatest around the dynamic discontinuities and the switching point, as are the errors, whilst the disturbance is least here).

We then planned a new "time-optimal" reference trajectory based on the same model dynamics, but with the available torques backed off according to equation (4.5). The closed-loop system, driven with this trajectory, tracked the path within the tolerance and with errors shown by the dashed lines in Figure 4.7 that closely resembled the predicted errors. Further, Figure 4.8 reveals that the actual compensation torques also closely resemble those predicted (Figure 4.6). Of course, we would not expect the predicted and actual errors and compensation torques to be identical because of the different disturbance processes acting for an identification experiment and the subsequent closed-loop experiment. However, they are clearly similar enough for this approach to be deemed useful. The normalised command torque plotted in Figure 4.9 indicates that the actuators did not saturate, and the average torque utilisation measure suggests approximately 75% utilisation. The elapsed time increased from $9.513s$ to $10.183s$.

It would be reasonable to stop at this point, having achieved the desired tracking accuracy with a small 7% increase in elapsed time and reasonable levels of torque utilisation. However, having obtained a solution which is closer to the "true" robust time-optimal solution than our initial "theoretical" time-optimal solution, we might now step through the process again, using the results of this experiment to re-identify $\ddot{\mathbf{q}}_d(t)$ and to choose more appropriate controller gains, compensation torques, and a reference trajectory for a second experiment. This might yield better results, i.e. solutions where the motors are more fully utilised, producing a faster tracking time, while still satisfying the tracking tolerance. Of course, we could continue to iterate the process in the hope that we might converge to yet better solutions.

Figures 4.10 to 4.13 summarise the results of 10 such iterations. Clearly , there are better solutions than that for the initial iteration. Also, we note that whilst the tracking errors violate the prescribed tolerances during the second iteration, the iterative process does converge, although the solutions oscillate in a region rather than converging to a point. This is not surprising since there are different disturbance processes acting between successive iterations.

From this example it appears that the proposed method works well to achieve robust sub-optimal tracking after one iteration, but that further iterations do yield better results.

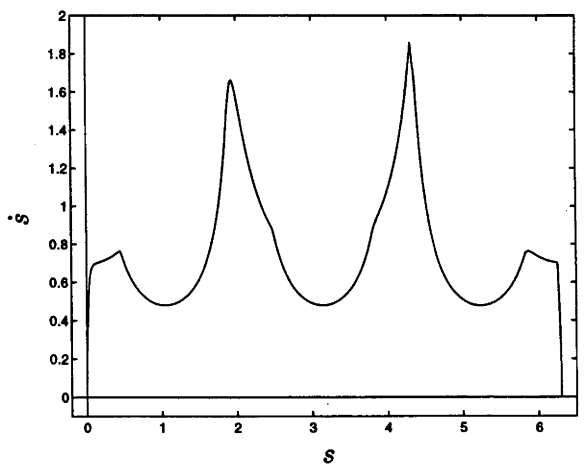


Figure 4.2: Phase Plane Plot of the Initial Trajectory

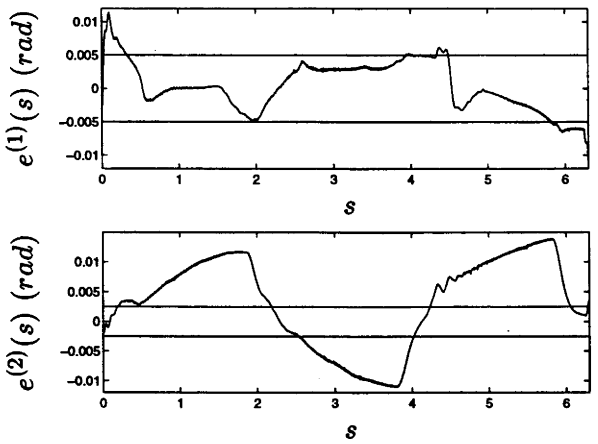


Figure 4.3: Tracking Errors - Initial Trajectory

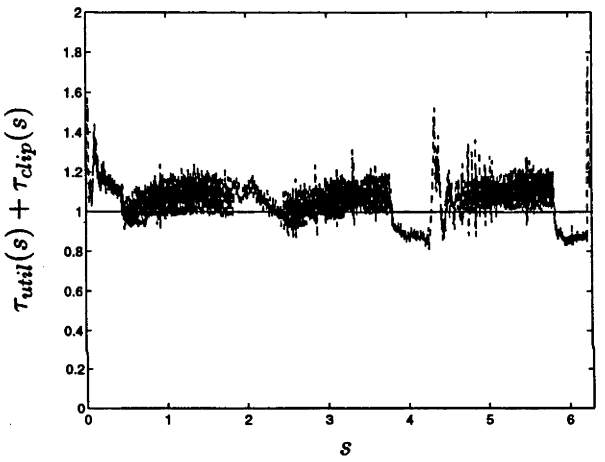


Figure 4.4: Normalised Command Torque - Initial Trajectory

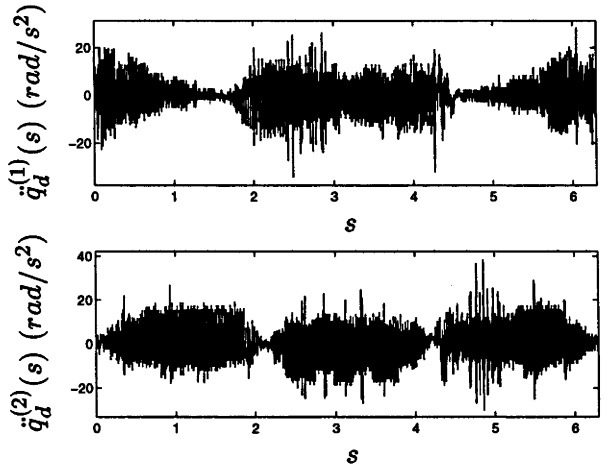


Figure 4.5: Acceleration Disturbances - Iteration 1

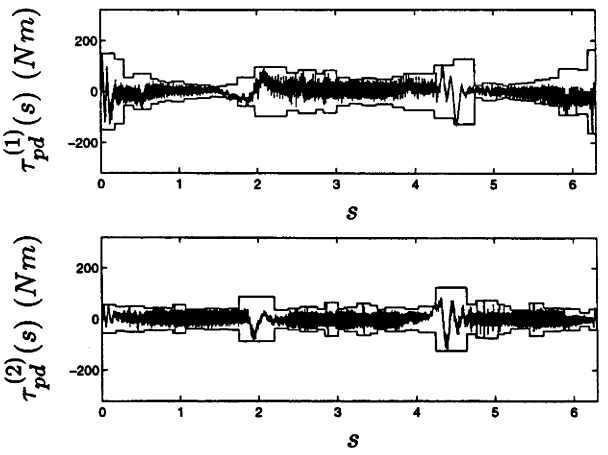


Figure 4.6: Predicted Compensation Torques and Bounds - Iteration 1

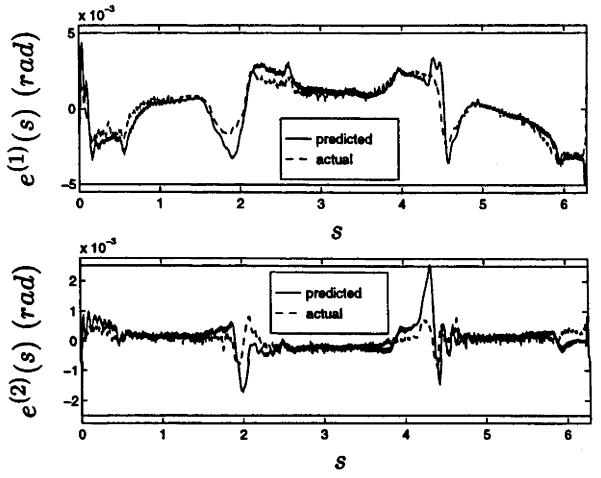


Figure 4.7: Predicted and Actual Tracking Errors - Iteration 1

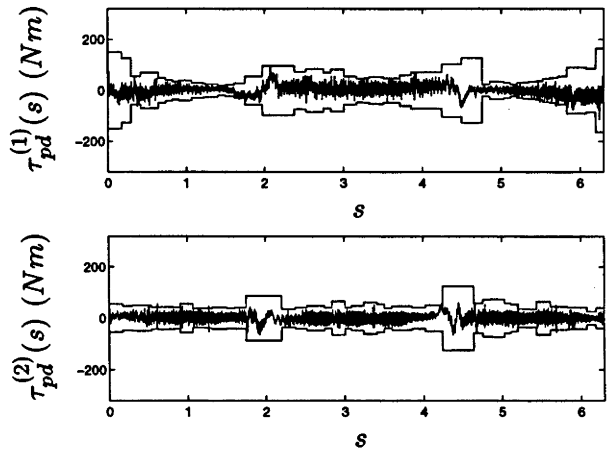


Figure 4.8: Actual Compensation Torques and Predicted Bounds - Iteration 1

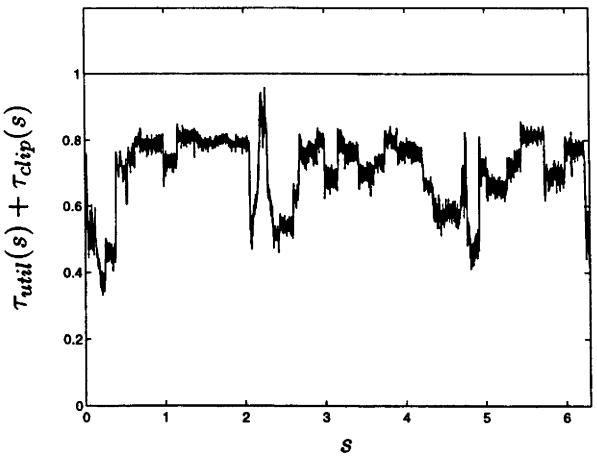


Figure 4.9: Normalised Command Torque - Iteration 1

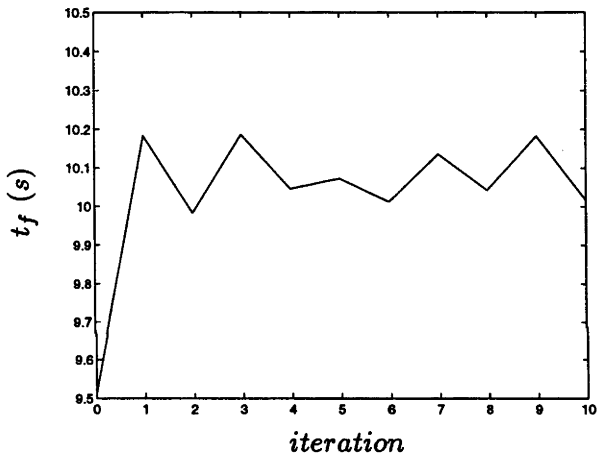


Figure 4.10: Tracking Time for 10 Iterations

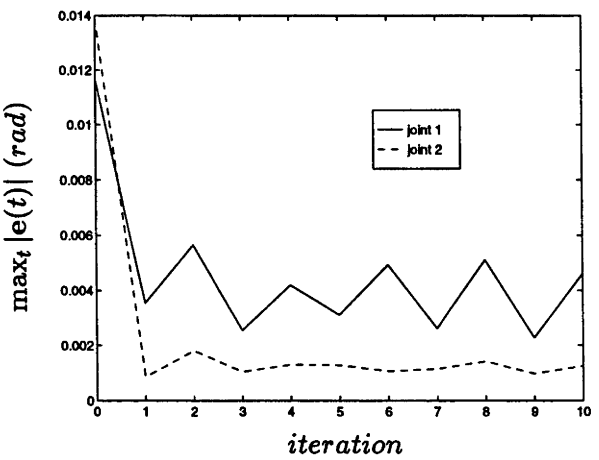


Figure 4.11: Maximum Error in Tracking For 10 Iterations

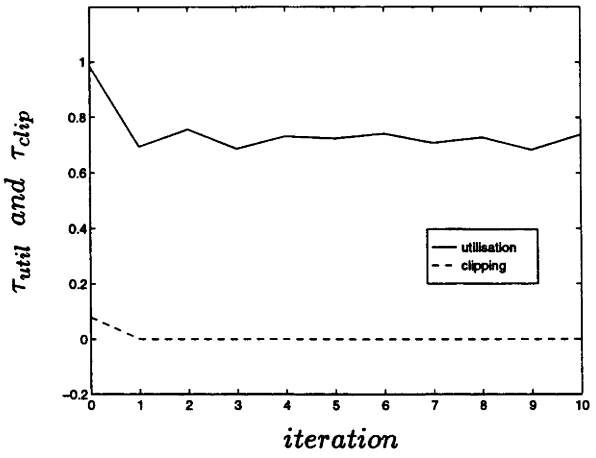


Figure 4.12: Torque Utilisation and Clipping For 10 Iterations

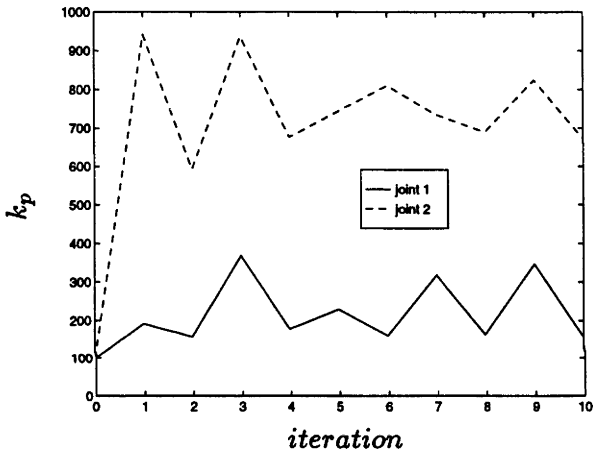


Figure 4.13: Proportional Gains For 10 Iterations

Iteration	Performance Measure						
	t_f (s)	$\max_t \|e(t)\|$ (rad)		τ_{util}	τ_{clip}	$\mathbf{k}_p$	
0	9.513	0.0117	0.0139	0.9838	0.0797	100.0	100.0
1	10.183	0.0035	0.0009	0.6939	0	191.4	942.2
2	9.983	0.0057	0.0018	0.7564	0.0001	157.4	592.6
3	10.187	0.0026	0.0010	0.6870	0.0003	368.0	936.9
4	10.047	0.0042	0.0013	0.7324	0.0000	177.8	676.2
5	10.073	0.0031	0.0013	0.7245	0.0000	229.2	743.8
6	10.013	0.0049	0.0011	0.7422	0.0001	160.7	810.3
7	10.137	0.0026	0.0011	0.7086	0.0000	318.1	735.3
8	10.043	0.0051	0.0014	0.7290	0.0000	164.0	689.9
9	10.183	0.0023	0.0010	0.6853	0.0000	346.9	824.6
10	10.017	0.0046	0.0013	0.7392	0.0000	157.4	671.9

Table 4.1: Robust Time-Optimal Trajectory Planning Results - Example Path

4.6 Robustness of the Method

Whilst the results for the example in § 4.5 were good, they do not guarantee that the method will work well for all achievable tasks (remember that there are tasks which are not achievable - either the disturbance accelerations are too high and/or the desired tolerance too small so that the compensation torques may be greater than those torques available). We wish to test the method's general utility as a "black box" solution. Thus, we wish to assess the robustness of the method over a much larger range of tasks.

To this end, we applied the method to 10 randomly generated paths. The paths were generated as two independent random n'th order polynomial functions of the path parameter s, one for each joint, using the following procedure:

1. Set the limits on the joint angle range.
2. Randomly select $n+1$ points that lay within the joint limits.
3. Associate a value of s with each point, with the s values spaced at equal intervals within the range $s \in [s_0, s_f]$.
4. Determine the n'th order polynomial such that the resulting parametric path passes through the $n+1$ points at the appointed values of s.
5. If points on the polynomial lay outside the joint limits, then rescale the polynomial such that all points in the range $s \in [s_0, s_f]$ lie within the joint limits (note that this will mean that the original points may not be part of the new polynomial).

Note that in our experimental system, § 4.3, we may store up to a maximum of $12s$ *of data. Thus, we do not wish that a time-optimal trajectory calculated using one of these parameterisations takes longer than* $12s$ *to complete. To prevent this, we simply limit the parameter range.*

The 10 paths produced by this procedure are displayed in the Cartesian space in Appendix 4.A. Note that the procedure produces paths which exercise a large proportion of the workspace and possess some very demanding features, which is what we desire.

For each experiment, we chose the tracking tolerance to be $\epsilon = (0.005, 0.005)^T\ rad$, which equates to approximately 5mm-6mm in the end-effector position. For the initial closed-loop experiment for each path, we set the values of the controller gains to be $\mathbf{k}_p = (100, 100)^T$. We iterated the procedure 9 times for each path. The results of all of our experiments are detailed in Tables 4.2 to 4.11 below.

The results are excellent. In all cases, the first iteration of the process provided a time-optimal trajectory which, when applied to the closed-loop system, yielded tracking errors within the specified tolerances.

The increases in elapsed time over the "theoretical" time-optimal ranged between 7% and 28%. The torque utilisation ranged from $\tau_{util} \approx 0.4$ and $\tau_{util} \approx 0.75$. In some cases the commanded torques were clipped, which is undesirable since it is when the system saturates that the ability to control the errors is lost, but not significantly enough to cause the tracking bounds to be violated.

Although for certain tasks the torque utilisation resulting from the first iteration of the process is low, the results might be still deemed acceptable. Such a decision would depend, in part, on how many times the manufacturing task is to be repeated - for example, to cut 20 parts or 20,000 ? For large manufacturing runs, an attempt to increase the torque utilisation/decrease the elapsed time might be deemed worthwhile. In such cases, further iterations might then yield better solutions.

Indeed, subsequent iterations of the process did produce better solutions, whose tracking errors still satisfied the specified tolerances but with significant increases in torque utilisation and reductions in elapsed time compared to those of the first iteration (for path 10, for example, the increase in the elapsed time was reduced from 28% compared to the nominal to approximately 10%). Also, the clipping of the command

torques was reduced to negligible levels, as desired. Whilst some solutions slightly violate the tracking tolerance, there are always adjacent solutions which do satisfy the criteria for little or no loss in time. (Of course, we would not expect the results to be exact since the additional noise processes acting during each iteration differ between successive iterations.) The method converged to some region in all cases. We have indicated the point at which we deem the method to have converged (which is subjective) by an asterisk (*) in the tables.

From all of these examples it appears that the proposed method works well to achieve robust sub-optimal tracking after one iteration, but that again, further iterations do yield better results.

Even with the increase in torque utilisation, the utilisation for certain examples remains relatively low (for example, for paths 5, 8 and 10). However, we would not expect this to be perfect because of the changing noise processes between iterations. In the next chapter, we will seek to address the issue of utilising more fully the available torques.

Iteration	Performance Measure						
	t_f (s)	$\max_t \lvert e(t) \rvert$ (rad)		τ_{util}	τ_{clip}	k_p	
0	9.597	0.0125	0.0090	0.9819	0.0768	100.0	100.0
1	10.290	0.0030	0.0035	0.7487	0.0004	268.8	157.4
2	9.960	0.0061	0.0035	0.8183	0.0006	139.4	148.2
3*	10.170	0.0042	0.0038	0.7690	0.0005	220.1	148.2
4	10.033	0.0053	0.0034	0.7930	0.0001	147.9	167.2
5	10.093	0.0044	0.0036	0.7837	0.0002	174.5	147.5
6	10.020	0.0049	0.0037	0.8011	0.0004	164.2	152.3
7	10.080	0.0043	0.0034	0.7858	0.0016	190.3	161.7
8	10.057	0.0051	0.0036	0.7830	0.0001	156.5	161.7
9	10.060	0.0047	0.0038	0.7891	0.0003	182.9	157.0

Table 4.2: Robust Time-Optimal Trajectory Planning Results - Random Path 1

Iteration	Performance Measure						
	t_f (s)	$\max_t \|e(t)\|$ (rad)		τ_{util}	τ_{clip}	k_p	
0	9.007	0.0164	0.0231	0.9927	0.1313	100.0	100.0
1	10.440	0.0024	0.0027	0.6151	0.0000	381.2	750.8
2	9.473	0.0051	0.0050	0.7888	0.0003	166.6	334.1
3*	9.863	0.0039	0.0034	0.7037	0.0002	218.7	589.3
4	9.640	0.0042	0.0041	0.7437	0.0000	192.6	444.4
5	9.797	0.0037	0.0035	0.7076	0.0000	217.8	515.6
6	9.623	0.0041	0.0039	0.7525	0.0000	204.9	389.2
7	9.693	0.0039	0.0046	0.7398	0.0000	211.1	389.2
8	9.743	0.0043	0.0036	0.7164	0.0000	198.5	533.3
9	9.777	0.0037	0.0043	0.7193	0.0000	236.1	500.6

Table 4.3: Robust Time-Optimal Trajectory Planning Results - Random Path 2

Iteration	Performance Measure						
	t_f (s)	$\max_t \|e(t)\|$ (rad)		τ_{util}	τ_{clip}	k_p	
0	9.633	0.0203	0.0141	0.9508	0.0777	100.0	100.0
1	10.757	0.0028	0.0016	0.5689	0.0000	325.0	789.1
2	10.290	0.0062	0.0046	0.6635	0.0001	167.5	375.2
3*	10.480	0.0040	0.0031	0.6200	0.0003	271.6	519.7
4	10.330	0.0050	0.0028	0.6573	0.0000	206.2	519.7
5	10.413	0.0041	0.0027	0.6273	0	231.0	589.7
6	10.353	0.0050	0.0031	0.6426	0	224.1	517.3
7	10.400	0.0038	0.0031	0.6386	0.0000	272.6	485.5
8	10.350	0.0050	0.0031	0.6473	0	223.3	485.5
9	10.410	0.0036	0.0027	0.6395	0.0000	271.9	556.1

Table 4.4: Robust Time-Optimal Trajectory Planning Results - Random Path 3

Iteration	Performance Measure						
	t_f (s)	$\max_t \|e(t)\|$ (rad)		τ_{util}	τ_{clip}	k_p	
0	9.513	0.0178	0.0136	0.9946	0.1146	100.0	100.0
1	10.350	0.0030	0.0035	0.6889	0	310.9	712.5
2	9.957	0.0055	0.0048	0.7911	0.0001	169.9	317.3
3*	10.120	0.0045	0.0031	0.7391	0.0000	208.8	536.8
4	10.033	0.0051	0.0039	0.7600	0.0001	196.4	438.1
5	10.153	0.0046	0.0034	0.7352	0.0001	234.0	545.0
6	10.093	0.0044	0.0031	0.7528	0.0001	234.0	444.7
7	10.087	0.0043	0.0036	0.7560	0	220.0	444.7
8	10.017	0.0051	0.0039	0.7620	0.0001	206.9	480.3
9	10.107	0.0040	0.0033	0.7432	0.0000	231.7	480.3

Table 4.5: Robust Time-Optimal Trajectory Planning Results - Random Path 4

Iteration	Performance Measure						
	t_f (s)	$\max_t \|e(t)\|$ (rad)		τ_{util}	τ_{clip}	$\mathbf{k}_p$	
0	7.633	0.0126	0.0117	0.8514	0.0093	100.0	100.0
1	9.417	0.0039	0.0019	0.4152	0.0004	325.0	635.9
2	8.750	0.0056	0.0071	0.4919	0.0004	236.4	176.3
3	8.933	0.0041	0.0029	0.4659	0.0003	284.1	364.7
4	8.770	0.0054	0.0039	0.4842	0.0001	232.7	253.9
5	8.747	0.0052	0.0057	0.4784	0.0001	244.7	223.4
6*	8.723	0.0044	0.0043	0.4830	0.0001	256.5	335.3
7	8.663	0.0050	0.0047	0.4970	0.0001	241.1	294.7
8	8.757	0.0045	0.0045	0.4885	0	264.8	294.7
9	8.690	0.0050	0.0044	0.4877	0.0001	248.9	276.9

Table 4.6: Robust Time-Optimal Trajectory Planning Results - Random Path 5

Iteration	Performance Measure						
	t_f (s)	$\max_t \|e(t)\|$ (rad)		τ_{util}	τ_{clip}	$\mathbf{k}_p$	
0	9.083	0.0118	0.0098	0.8922	0.0320	100.0	100.0
1	10.410	0.0032	0.0025	0.4985	0	339.1	463.7
2	9.873	0.0060	0.0061	0.6002	0.0004	195.1	165.9
3*	10.050	0.0044	0.0045	0.5743	0.0000	258.0	241.5
4	9.893	0.0054	0.0050	0.5950	0.0003	227.0	212.5
5	10.043	0.0043	0.0043	0.5730	0.0000	263.2	231.2
6	9.857	0.0053	0.0055	0.5975	0.0000	223.7	203.6
7	10.063	0.0045	0.0034	0.5692	0.0000	260.0	259.8
8	9.897	0.0060	0.0059	0.5922	0.0001	221.0	181.7
9	10.053	0.0046	0.0038	0.5763	0.0000	257.5	257.0

Table 4.7: Robust Time-Optimal Trajectory Planning Results - Random Path 6

Iteration	Performance Measure						
	t_f (s)	$\max_t \|e(t)\|$ (rad)		τ_{util}	τ_{clip}	$\mathbf{k}_p$	
0	9.283	0.0125	0.0148	0.8833	0.0220	100.0	100.0
1	11.000	0.0041	0.0012	0.4624	0	353.1	1018.8
2	10.083	0.0055	0.0071	0.5865	0.0006	213.7	167.6
3*	10.297	0.0045	0.0023	0.5342	0.0001	250.6	771.7
4	10.247	0.0051	0.0048	0.5619	0.0001	235.6	248.0
5	10.210	0.0054	0.0028	0.5451	0.0000	247.5	693.5
6	10.337	0.0064	0.0042	0.5474	0.0001	271.0	287.7
7	10.253	0.0045	0.0025	0.5337	0	254.7	655.8
8	10.263	0.0048	0.0041	0.5584	0.0000	239.4	292.6
9	10.193	0.0048	0.0028	0.5461	0	239.4	660.3

Table 4.8: Robust Time-Optimal Trajectory Planning Results - Random Path 7

Iteration	Performance Measure						
	t_f (s)	$\max_t \|e(t)\|$ (rad)	τ_{util}	τ_{clip}	k_p		
0	9.670	0.0128	0.0060	0.8578	0.0063	100.0	100.0
1	11.527	0.0046	0.0025	0.4225	0.0002	493.8	186.1
2	10.637	0.0054	0.0056	0.5144	0.0011	229.2	87.1
3*	10.590	0.0049	0.0031	0.5183	0.0000	265.3	183.0
4	10.607	0.0059	0.0029	0.5063	0.0001	265.3	150.6
5	10.557	0.0052	0.0046	0.5256	0.0003	249.4	93.5
6	10.580	0.0054	0.0032	0.5228	0.0001	261.1	141.4
7	10.603	0.0048	0.0050	0.5139	0.0003	284.2	83.9
8	10.550	0.0052	0.0048	0.5230	0.0008	258.5	83.9
9	10.600	0.0051	0.0040	0.5163	0.0001	270.1	103.1

Table 4.9: Robust Time-Optimal Trajectory Planning Results - Random Path 8

Iteration	Performance Measure						
	t_f (s)	$\max_t \|e(t)\|$ (rad)	τ_{util}	τ_{clip}	k_p		
0	9.280	0.0152	0.0285	0.9076	0.0774	100.0	100.0
1	10.963	0.0031	0.0046	0.4955	0.0009	423.4	559.4
2	10.020	0.0065	0.0029	0.6082	0.0005	223.2	422.0
3	10.270	0.0044	0.0064	0.5971	0.0007	320.3	241.8
4*	10.150	0.0051	0.0028	0.5828	0.0002	281.5	464.8
5	10.173	0.0042	0.0051	0.5962	0.0000	303.9	237.4
6	10.103	0.0056	0.0034	0.5937	0.0002	258.0	423.4
7	10.197	0.0045	0.0040	0.5907	0	304.4	268.4
8	10.063	0.0054	0.0045	0.5983	0	276.8	286.9
9	10.157	0.0045	0.0031	0.5816	0	310.7	434.2

Table 4.10: Robust Time-Optimal Trajectory Planning Results - Random Path 9

Iteration	Performance Measure						
	t_f (s)	$\max_t \|e(t)\|$ (rad)	τ_{util}	τ_{clip}	k_p		
0	9.213	0.0145	0.0104	0.8686	0.0265	100.0	100.0
1	11.817	0.0031	0.0049	0.3912	0.0021	423.4	1018.8
2	10.543	0.0066	0.0042	0.4682	0.0012	229.6	892.7
3*	10.440	0.0045	0.0024	0.4705	0.0002	325.9	782.3
4	10.297	0.0048	0.0050	0.5066	0.0002	296.3	251.4
5	10.473	0.0045	0.0019	0.4850	0.0001	278.4	844.9
6	10.370	0.0054	0.0049	0.5093	0.0003	261.6	349.2
7	10.363	0.0047	0.0019	0.4793	0.0002	284.7	930.5
8	10.307	0.0055	0.0056	0.5216	0.0004	267.5	297.7
9	10.467	0.0046	0.0019	0.4777	0.0000	301.9	885.5

Table 4.11: Robust Time-Optimal Trajectory Planning Results - Random Path 10

4.7 Technical Note

The random paths that we used caused large forces to be exerted on the robot. As we stepped through the experiments, these forces caused substantial backlash to develop in the elbow joint. The implication of this can be seen clearly in the results tables, where the performance in terms of both the maximum errors and torque utilisation degraded significantly for the later paths. It is interesting to note that even in the presence of the backlash, the method still proffers useful results.

After having fixed the backlash, we decided that it might be useful to perform some of the experiments again, primarily to confirm the truth of our assumption that the degradation in performance was due to the backlash. We decided not to perform all of the experiments, for fear of provoking further backlash or potentially more serious damage, instead limiting ourselves to performing experiments for two of the random paths which were most visibly affected by the backlash; path 6 which had many outlying points prior to the backlash being fixed, and path 10 which previously had relatively poor torque utilisation.

As part of this action, it came to our attention that the removal of the backlash had caused the dynamics of the robot to change significantly enough to make the model parameters quite poor. This made it necessary to re-identify the parameters prior to running the examples again. The new parameters are shown in Table 4.12.

Because of the change in the model parameters, and because of our desire to compare results from this scheme with results from an extension to this scheme which we will present in the next chapter, we also chose to run the process again for the example path of § 4.5 and for random path 2, both of which we will revisit in the next chapter.

The results from the example path, and from random paths 2, 6 and 10 are displayed in Tables 4.13, 4.14, 4.15 and 4.16 respectively.

The results for the example path are very similar here to before, *c.f.* Table 4.1. The torque utilisation is almost the same, although the tracking errors are a little more

Parameter	Value	Units	Represents
f_{11}	1.0456	Nms^2/rad	Inertia terms
$f_{12} = f_{21}$	0.5228	Nms^2/rad	
g_{11}	2.7238	Nms^2/rad	
$g_{12} = g_{21}$	1.3619	Nms^2/rad	
h_{11}	30.3078	Nms^2/rad	
$h_{12} = h_{21}$	2.5433	Nms^2/rad	
h_{22}	14.8486	Nms^2/rad	
u_1	-2.7238	Nms^2/rad^2	Centrifugal
u_2	1.3619	Nms^2/rad^2	and coriolis
v_1	-1.3619	Nms^2/rad^2	terms
w_1	1.0456	Nms^2/rad^2	
w_2	-0.5228	Nms^2/rad^2	
x_1	0.5228	Nms^2/rad^2	
f_{1p}	16.1108	Nm	Friction terms
f_{1n}	-12.8595	Nm	
f_{2p}	12.3031	Nm	
f_{2n}	-12.6793	Nm	
b_{1p}	23.6082	Nms/rad	
b_{1n}	29.7581	Nms/rad	
b_{2p}	20.7015	Nms/rad	
b_{2n}	18.3675	Nms/rad	
l_1	0.6100	m	Length of links
l_2	0.5800	m	1 and 2

Table 4.12: Re-Identified SCARA Manipulator Parameter Data

tightly constrained (there are no outlying points). This is not wholly surprising since the backlash was only beginning to develop when we ran the original experiments for this path. Similarly, the results for random path 2 are close to the original results with only a small increase in the torque utilisation, *c.f.* Table 4.3.

However, the results for random paths 6 and 10 are much better, in terms of both outlying points and torque utilisation, *c.f.* Tables 4.7 and 4.11. There is now only one outlying point (which occurs in the second iteration for path 6) and the utilisation has increased from $\tau_{util} \sim O(0.58)$ to $\sim O(0.72)$ for path 6, and from $\tau_{util} \sim O(0.5)$ to $\sim O(0.65)$ for path 10.

It is clear that without the effect of backlash, the results we obtain are excellent, with fast convergence, minimal outlying points, and significantly higher torque utilisation.

Iteration	Performance Measure						
	t_f (s)	$\max_t \|e(t)\|$ (rad)		τ_{util}	τ_{clip}	k_p	
0	9.487	0.0151	0.0129	0.9815	0.0734	100.0	100.0
1	10.153	0.0036	0.0010	0.6910	0.0002	465.6	1095.3
2	9.927	0.0043	0.0017	0.7645	0.0001	156.2	655.1
3	10.043	0.0028	0.0015	0.7250	0.0000	325.0	942.2
4	9.977	0.0041	0.0014	0.7443	0.0000	170.3	903.9
5	10.007	0.0033	0.0016	0.7423	0.0002	296.9	712.5
6	9.997	0.0041	0.0013	0.7326	0	198.4	942.2
7	10.030	0.0034	0.0014	0.7241	0	268.8	827.3
8	9.990	0.0036	0.0013	0.7390	0	226.6	865.6
9	9.983	0.0034	0.0016	0.7480	0.0001	254.7	827.3

Table 4.13: Robust Time-Optimal Trajectory Planning Results with Backlash Removed - Example Path

Iteration	Performance Measure						
	t_f (s)	$\max_t \|e(t)\|$ (rad)		τ_{util}	τ_{clip}	k_p	
0	8.910	0.0139	0.0228	0.9916	0.1153	100.0	100.0
1	10.337	0.0020	0.0020	0.6343	0.0001	606.2	827.3
2	9.303	0.0056	0.0061	0.8272	0.0012	170.3	291.4
3	9.857	0.0028	0.0024	0.6886	0.0000	268.8	789.1
4	9.410	0.0048	0.0053	0.7752	0.0009	170.3	482.8
5	9.757	0.0028	0.0038	0.6977	0.0006	240.6	789.1
6	9.570	0.0045	0.0034	0.7358	0.0001	163.3	712.5
7	9.443	0.0043	0.0046	0.7785	0.0004	184.4	406.2
8	9.587	0.0031	0.0037	0.7405	0.0000	240.6	559.4
9	9.537	0.0048	0.0047	0.7515	0.0003	170.3	559.4

Table 4.14: Robust Time-Optimal Trajectory Planning Results with Backlash Removed - Random Path 2

Iteration	Performance Measure						
	t_f (s)	$\max_t \|e(t)\|$ (rad)		τ_{util}	τ_{clip}	k_p	
0	8.903	0.0176	0.0131	0.9744	0.0690	100.0	100.0
1	10.253	0.0027	0.0022	0.5799	0.0003	493.8	348.8
2	9.390	0.0057	0.0048	0.7466	0.0008	156.2	186.1
3	9.700	0.0034	0.0045	0.6620	0.0001	296.9	243.6
4	9.493	0.0047	0.0030	0.7144	0.0000	184.4	329.7
5	9.533	0.0037	0.0046	0.7070	0.0002	240.6	234.0
6	9.467	0.0047	0.0028	0.7226	0.0001	184.4	329.7
7	9.533	0.0037	0.0041	0.7059	0.0001	226.6	253.1
8	9.470	0.0040	0.0033	0.7172	0.0001	212.5	291.4
9	9.470	0.0039	0.0044	0.7266	0.0003	212.5	234.0

Table 4.15: Robust Time-Optimal Trajectory Planning Results with Backlash Removed - Random Path 6

Iteration	Performance Measure						
	t_f (s)	$\max_t \|e(t)\|$ (rad)		τ_{util}	τ_{clip}	k_p	
0	9.197	0.0138	0.0119	0.9672	0.0663	100.0	100.0
1	10.473	0.0030	0.0016	0.5630	0.0010	381.2	1171.9
2	9.877	0.0043	0.0050	0.6707	0.0003	184.4	214.8
3	10.010	0.0037	0.0013	0.6270	0.0001	240.6	1018.8
4	9.847	0.0040	0.0050	0.6801	0.0001	198.4	291.4
5	10.007	0.0041	0.0015	0.6396	0.0001	212.5	1095.3
6	9.903	0.0036	0.0048	0.6683	0.0000	226.6	272.3
7	9.917	0.0041	0.0025	0.6564	0.0005	212.5	1018.8
8	9.953	0.0039	0.0024	0.6483	0.0002	240.6	482.8
9	9.883	0.0038	0.0031	0.6733	0.0001	226.6	482.8

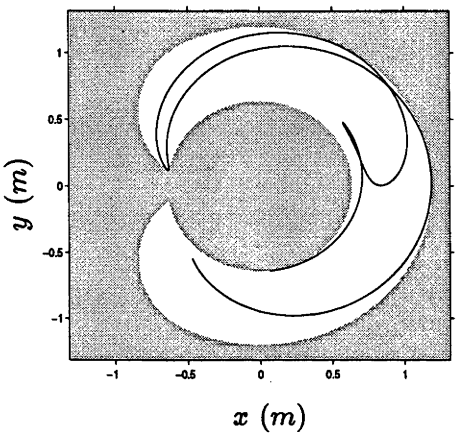
Table 4.16: Robust Time-Optimal Trajectory Planning Results with Backlash Removed - Random Path 10

4.8 Conclusions

We have proposed a method which uses the theory developed in Chapter 3 to identify the robot modelling errors as disturbances in joint accelerations, and to use these to predict the torque levels and controller gains required to compensate for these disturbances and so to enable the planning of a robust time-optimal trajectory. Experimental results confirm that this theory can be applied in practice with excellent results.

Appendix

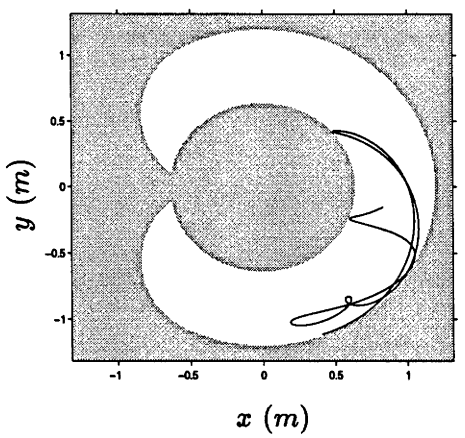
4.A Robust Trajectory Planning - Random Paths



Order 7, $s \in [3.000, 6.283]$

	q_1	q_2
a_0	-1.395e+00	2.372e-01
a_1	3.642e+00	-1.677e+01
a_2	6.736e+00	4.185e+01
a_3	-1.486e+01	-3.783e+01
a_4	9.047e+00	1.639e+01
a_5	-2.453e+00	-3.712e+00
a_6	3.112e-01	4.240e-01
a_7	-1.507e-01	-1.928e-02
limits	2.000	2.000

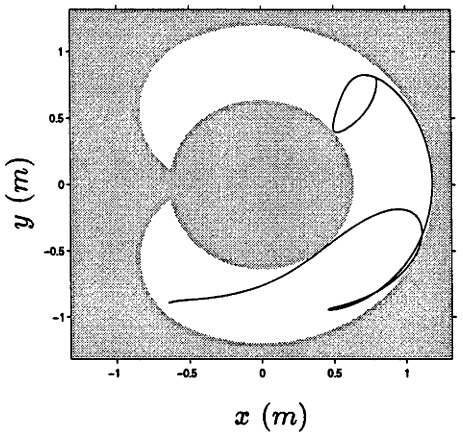
Figure 4.14: Cartesian Space Path and Generating Path Data - Path 1



Order 9, $s \in [0.500, 5.500]$

	q_1	q_2
a_0	-5.301e-01	-9.206e-01
a_1	-1.680e+01	-1.497e+01
a_2	6.348e+01	6.322e+01
a_3	-9.408e+01	-8.134e+01
a_4	7.286e+01	4.981e+01
a_5	-3.270e+01	-1.660e+01
a_6	8.799e+00	3.089e+00
a_7	-1.399e+00	-3.012e-01
a_8	1.209e-01	1.199e-02
a_9	-4.379e-03	-1.690e-14
limits	2.000	2.000

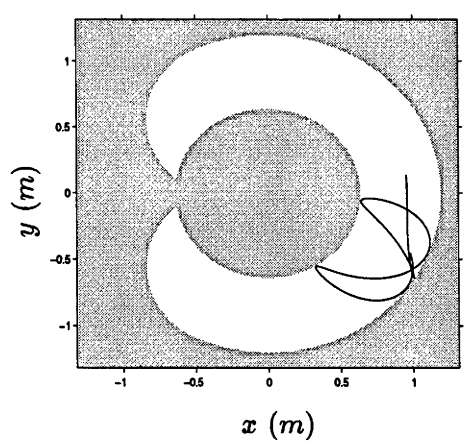
Figure 4.15: Cartesian Space Path and Generating Path Data - Path 2



Order 7, $s \in [1.500, 6.283]$

	q_1	q_2
a_0	4.900e-01	6.326e-01
a_1	1.438e+01	-2.781e+00
a_2	-4.234e+01	1.165e+01
a_3	3.996e+01	-1.550e+01
a_4	-1.750e+01	8.400e+00
a_5	3.914e+00	-2.167e+00
a_6	-4.344e-01	2.670e-01
a_7	1.898e-02	-1.266e-02
limits	2.000	2.000

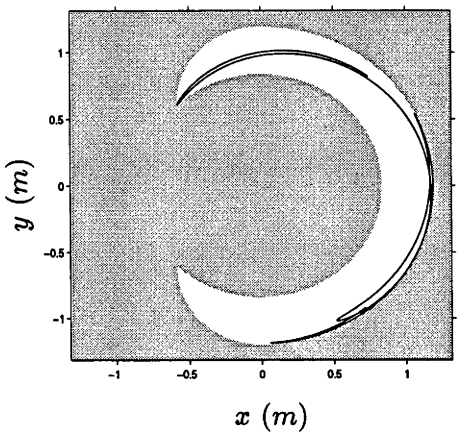
Figure 4.16: Cartesian Space Path and Generating Path Data - Path 3



Order 5, $s \in [1.800, 6.200]$

	q_1	q_2
a_0	-7.997e-01	5.530e-01
a_1	1.086e+01	-7.072e+00
a_2	-1.350e+01	1.101e+01
a_3	5.911e+00	-5.545e+00
a_4	-1.069e+00	1.093e+00
a_5	6.806e-02	-7.357e-02
limits	2.000	2.000

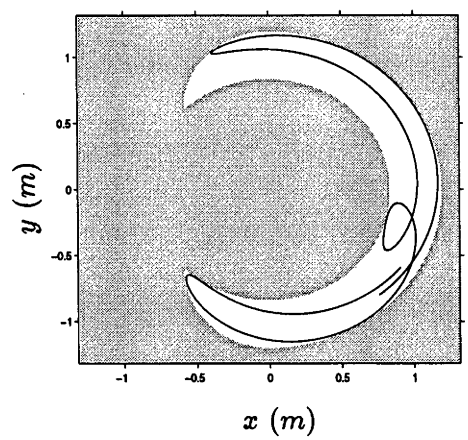
Figure 4.17: Cartesian Space Path and Generating Path Data - Path 4



Order 5, $s \in [0.000, 6.283]$

	q_1	q_2
a_0	4.648e-01	7.624e-01
a_1	5.157e+00	3.476e+00
a_2	-7.392e+00	-4.556e+00
a_3	3.285e+00	1.801e+00
a_4	-5.842e-01	-2.899e-01
a_5	3.610e-02	1.662e-02
limits	1.5708	1.5708

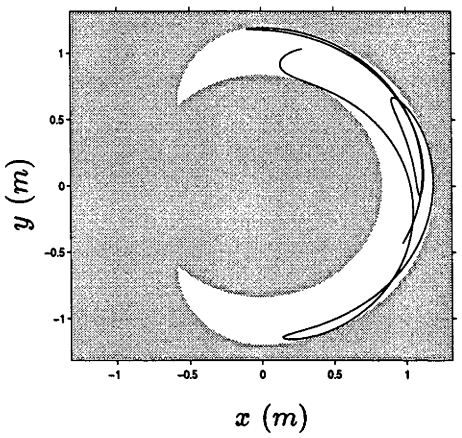
Figure 4.18: Cartesian Space Path and Generating Path Data - Path 5



Order 5, $s \in [0.000, 6.283]$

	q_1	q_2
a_0	-1.605e-01	-8.609e-01
a_1	-5.109e+00	-3.573e+00
a_2	6.114e+00	5.359e+00
a_3	-2.771e+00	-2.215e+00
a_4	5.287e-01	3.728e-01
a_5	-3.534e-02	-2.253e-02
limits	1.5708	1.5708

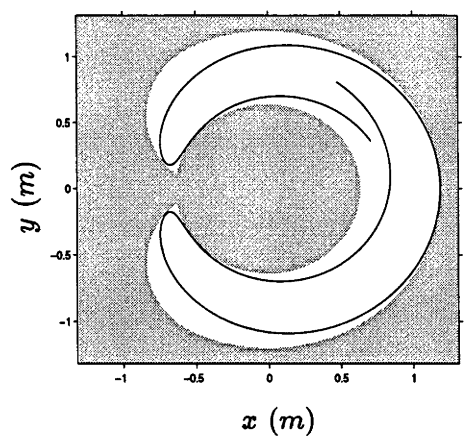
Figure 4.19: Cartesian Space Path and Generating Path Data - Path 6



Order 7, $s \in [0.000, 5.500]$

	q_1	q_2
a_0	8.629e-01	9.067e-01
a_1	-9.640e-02	6.572e+00
a_2	-1.120e+01	-1.985e+01
a_3	1.588e+01	1.834e+01
a_4	-8.716e+00	-7.901e+00
a_5	2.297e+00	1.758e+00
a_6	-2.906e-01	-1.969e-01
a_7	1.417e-02	8.783e-03
limits	1.5708	1.5708

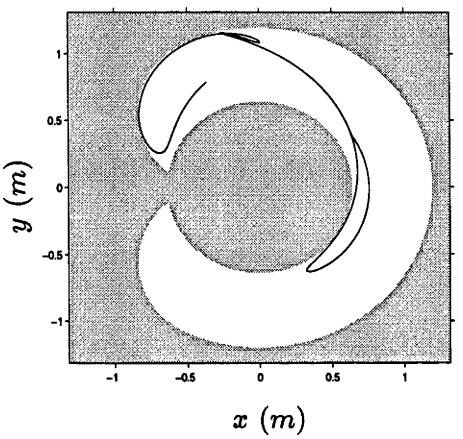
Figure 4.20: Cartesian Space Path and Generating Path Data - Path 7



Order 3, $s \in [0.000, 6.000]$

	q_1	q_2
a_0	1.683e+00	-1.330e+00
a_1	-5.358e+00	-1.482e+00
a_2	2.303e+00	9.098e-01
a_3	-2.443e-01	-9.653e-02
limits	2.000	2.000

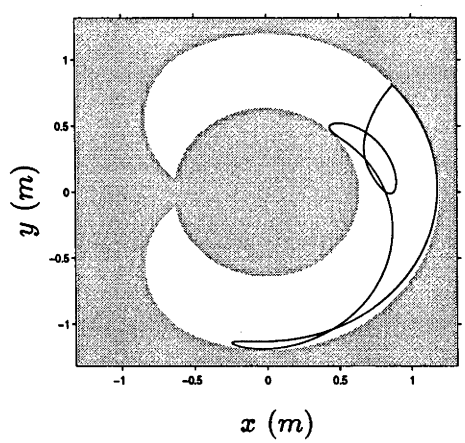
Figure 4.21: Cartesian Space Path and Generating Path Data - Path 8



Order 6, $s \in [0.000, 4.700]$

	q_1	q_2
a_0	-3.133e-01	1.795e+00
a_1	-9.801e+00	-7.490e-01
a_2	1.790e+01	4.264e+00
a_3	-1.105e+01	-5.394e+00
a_4	3.207e+00	2.310e+00
a_5	-4.446e-01	-4.034e-01
a_6	2.358e-02	2.478e-02
limits	2.000	2.000

Figure 4.22: Cartesian Space Path and Generating Path Data - Path 9



Order 8, $s \in [0.000, 3.700]$

	q_1	q_2
a_0	7.864e-01	-5.666e-02
a_1	-2.583e+01	9.620e+00
a_2	7.816e+01	-4.051e+01
a_3	-8.942e+01	5.298e+01
a_4	5.191e+01	-3.361e+01
a_5	-1.682e+01	1.162e+01
a_6	3.078e+00	-2.234e+00
a_7	-2.973e-01	2.238e-01
a_8	1.177e-02	-9.105e-03
limits	2.000	2.000

Figure 4.23: Cartesian Space Path and Generating Path Data - Path 10

Chapter 5

Closed-Loop Trajectory Generation for Robust Time-Optimal Path Tracking

Abstract

In the previous chapter, we developed a method for planning robust "time-optimal" reference trajectories for robots under computed-torque control. The scheme was based on the idea of holding enough torque on reserve during trajectory planning to ensure that the actuators would not saturate due to disturbances. This, together with the correct controller gain settings, ensured that the tracking was accurate to a specified tolerance. In this chapter, we extend that approach to a less conservative "closed-loop" architecture for "on-line trajectory generation" which is similar to Dahl and Nielsen's scheme for "on-line trajectory time-scaling". We propose a feedback control law for setting the reference path acceleration such that path tracking is nearly time-optimal, robustly controllable, and accurate to a prescribed tolerance. We show how this law can be determined off-line using shooting methods and implemented with little on-line memory. Experimental results demonstrate that the tracking times are reduced compared to our previous approach, while the tracking accuracy and robustness remain approximately the same.

5.1 Introduction

In Chapter 4 we proposed a scheme for planning robust "time-optimal" trajectories for robots under computed-torque control. This scheme is based on the experimental identification of the unmodelled dynamics as joint acceleration disturbances and a theory for relating such disturbances to the tracking performance of a robot under computed-torque control. The theory allows identification of the torque margins that need to be held in reserve and the controller gains that are required to reject the expected levels of disturbance, so that tracking is accurate to a prescribed tolerance.

We note, however, that the compensation torques with which we define the "robust" time-optimal trajectory are "worst-case". This means that if the on-line disturbance is less than this upper figure, then the torques will not be being fully utilised, i.e. there will be room for improvement. Certainly, the example in § 4.5 would suggest that this is the case since the actual disturbance is well inside the disturbance "envelope" for much of the time (*c.f.* Figure 4.8), and where the average utilisation measure is approximately 0.75 (versus a possible maximum value of 1.0 when at least one of the motors is always operating at its full potential).

In this chapter, we extend our previous approach to a more general architecture for on-line trajectory generation, similar to that of Dahl and Nielsen [22, 23, 24]. In this architecture, shown in Figure 5.1, the on-line *trajectory integrator* produces a reference joint trajectory $\mathbf{q}_r(t)$ in response to an input path acceleration $\ddot{s}$. Note that $\mathbf{q}_r(t) = \mathbf{f}(s(t))$ where $\mathbf{f}(s)$ denotes the prescribed path parameterised by s. Because $\ddot{s}$ is generated on-line as a feedback function of the plant and trajectory generator states, we call this a *closed-loop trajectory generator* (in contrast to the trajectory generation used in Chapter 4 which is open-loop). Dahl and Nielsen's scheme differs from this in that they use a dynamic control law in place of the state feedback function $u(s, \dot{s}, \mathbf{q}, \dot{\mathbf{q}})$.

The main contribution of this work is a proposal for a state feedback law $u(s, \dot{s}, \mathbf{q}, \dot{\mathbf{q}})$ that ensures near time-optimal path tracking accurate to a prescribed tolerance, and robust to experimentally identified levels of disturbance. This control law takes the

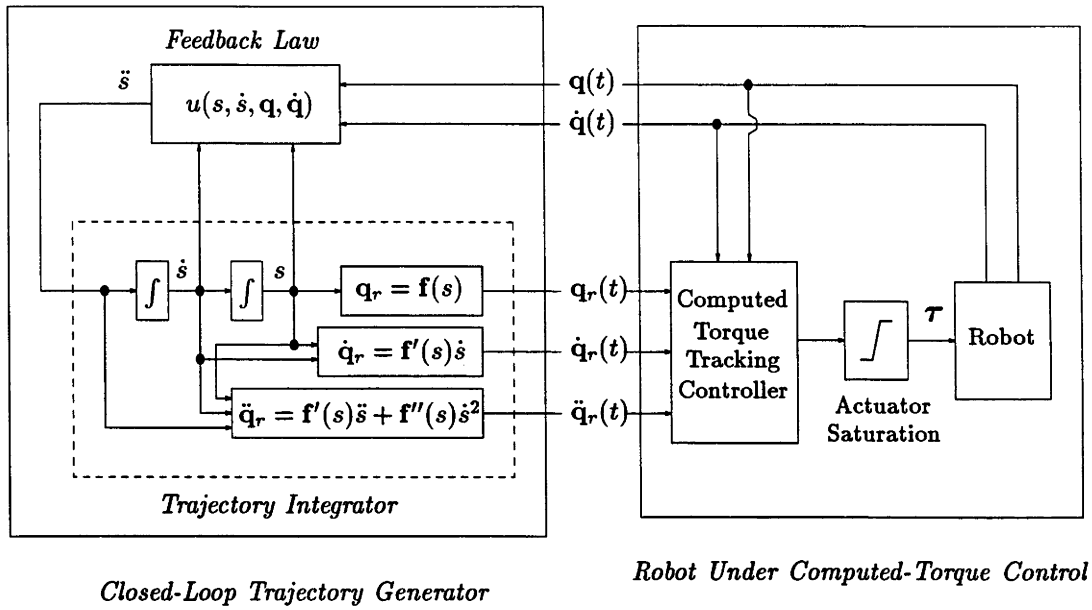


Figure 5.1: Closed-Loop Architecture for On-Line Trajectory Generation

form

$$\begin{aligned} \ddot{s} &= u(s, \dot{s}, \mathbf{q}, \dot{\mathbf{q}}) \\ &= sat(\ddot{\sigma}; \underline{\ddot{s}}, \overline{\ddot{s}}) \end{aligned}$$

where

$$\begin{aligned} \ddot{\sigma} &= \ddot{\sigma}(s, \dot{s}) \\ \underline{\ddot{s}} &= \underline{\ddot{s}}(s, \dot{s}, \mathbf{q}, \dot{\mathbf{q}}) \\ \overline{\ddot{s}} &= \overline{\ddot{s}}(s, \dot{s}, \mathbf{q}, \dot{\mathbf{q}}) \end{aligned}$$

Here, $\ddot{\sigma}(s, \dot{s})$ represents a *path acceleration policy* that is planned off-line and implemented on-line subject to saturation limits $\underline{\ddot{s}}$ and $\overline{\ddot{s}}$.

The limits $\underline{\ddot{s}}$ and $\overline{\ddot{s}}$ are computed on-line and represent bounds on the values of $\ddot{s}$ that may be imposed at the current instant without over-saturating any of the joint actuators. If either $\ddot{s} = \underline{\ddot{s}}$ or $\ddot{s} = \overline{\ddot{s}}$ is applied, then at least one actuator will be exactly saturated and none will be over-saturated.

The planning of $\ddot{\sigma}(s, \dot{s})$ occurs off-line in the $(s, \dot{s})$ phase plane. It involves the determination of a phase plane region C_{wc} that is controllable under worst-case disturbance

assumptions. The acceleration policy $\ddot{\sigma}(s, \dot{s})$ is simply to seek the boundary of C_{wc} and then to track it.

In § 5.2 we present our new approach. We develop equations for computing $\underline{\ddot{s}}$ and $\overline{\ddot{s}}$ and for making off-line predictions for these on-line bounds. We define worst-case admissible, worst-case controllable, and worst-case unreachable regions of the phase plane and present an algorithm for computing their boundaries. We then present and justify the proposed feedback control law. In § 5.3 we discuss the application of the new approach, and in § 5.4 we re-visit our experimental system and the performance measures for the experiments. In § 5.5 we present some experimental results which illustrate the method. In § 5.6 we demonstrate the robustness of the method by applying the method to several paths and compare the results with the previous method developed in Chapter 4. Finally, in § 5.7 we draw conclusions.

5.2 Robust Closed-Loop Trajectory Generation

5.2.1 Path Acceleration Bounds

To extend the method of Chapter 4 to the architecture shown in Figure 5.1, we need to determine how much the path acceleration $\ddot{s}$ can be modulated on-line without saturating any actuators. This is important because the system (3.4) and equation (3.5), which predict the tracking error and compensation torques, are only valid if the torque commands (3.3) are implemented without clipping.

We assume joint torque limits of the form

$$\underline{\tau}^{(i)}(\mathbf{q}, \dot{\mathbf{q}}) \leq \tau^{(i)} \leq \overline{\tau}^{(i)}(\mathbf{q}, \dot{\mathbf{q}}) , \qquad 1 \leq i \leq n . \tag{5.1}$$

The constraints (5.1) then give rise to on-line bounds for $\ddot{s}$ of the form

$$\underline{\ddot{s}}(s, \dot{s}, \mathbf{q}, \dot{\mathbf{q}}) \leq \ddot{s} \leq \overline{\ddot{s}}(s, \dot{s}, \mathbf{q}, \dot{\mathbf{q}}) . \tag{5.2}$$

Explicit expressions for $\underline{\ddot{s}}(\cdot)$ and $\overline{\ddot{s}}(\cdot)$ are derived in Appendix 5.A based on the substitution of (3.3) into (5.1) with $\mathbf{q}_r(t) = \mathbf{f}(s(t))$.

Further, assuming that the compensation torques generated on-line will be bounded by equations of the form

$$\underline{\tau}_{pd}(s) \leq \tau_{pd}(s) \leq \overline{\tau}_{pd}(s) \ , \tag{5.3}$$

and assuming that $\mathbf{q}(t) \approx \mathbf{f}(s(t))$, then bounds on $\underline{\ddot{s}}$ and $\overline{\ddot{s}}$ can be predicted off-line:

$$\underline{\ddot{s}}_{min}(s,\dot{s}) \leq \underline{\ddot{s}} \leq \underline{\ddot{s}}_{max}(s,\dot{s}) \ , \tag{5.4}$$

$$\overline{\ddot{s}}_{min}(s,\dot{s}) \leq \overline{\ddot{s}} \leq \overline{\ddot{s}}_{max}(s,\dot{s}) \ . \tag{5.5}$$

The inequalities (5.4) and (5.5) are derived in Appendix 5.B, including explicit expressions for $\underline{\ddot{s}}_{min}(s,\dot{s})$, $\underline{\ddot{s}}_{max}(s,\dot{s})$, $\overline{\ddot{s}}_{min}(s,\dot{s})$ and $\overline{\ddot{s}}_{max}(s,\dot{s})$.

Because these bounds are functions of s and $\dot{s}$ only, we can use them to plan the control of $\ddot{s}$ off-line in the $(s,\dot{s})$ plane. Furthermore, we can plan the control of $\ddot{s}$ knowing that its on-line range will include the *worst case* interval

$$\underline{\ddot{s}}_{max}(s,\dot{s}) \leq \ddot{s} \leq \overline{\ddot{s}}_{min}(s,\dot{s}) \ . \tag{5.6}$$

5.2.2 Worst-Case Admissible, Controllable, and Unreachable Regions

The proposed control scheme is based on the idea of taking advantage of the available path acceleration $\ddot{s}$ that develops on-line to generate the fastest trajectory that is *worst-case controllable.* By worst-case controllable we mean that it must be possible to reach the goal state from any point on the trajectory using $\ddot{s}$ restricted to the worst-case range (5.6).

Let the worst-case controllable region $\mathcal{C}_{wc}$ be defined as the set of phase plane states that can reach the goal state under worst-case admissible $\ddot{s}$ defined by (5.6). Figure 5.2 shows $\mathcal{C}_{wc}$ for a typical case and is intended to illustrate the arguments below.

As an aid to determining $\mathcal{C}_{wc}$ it is helpful to define the worst-case admissible region $\mathcal{A}_{wc}$. Let $\mathcal{A}_{wc}$ denote the phase plane region where inequalities (5.6) provide at least one admissible value for $\ddot{s}$, i.e.

$$\mathcal{A}_{wc} = \{(s,\dot{s}) : \underline{\ddot{s}}_{max}(s,\dot{s}) \leq \overline{\ddot{s}}_{min}(s,\dot{s})\} \ . \tag{5.7}$$

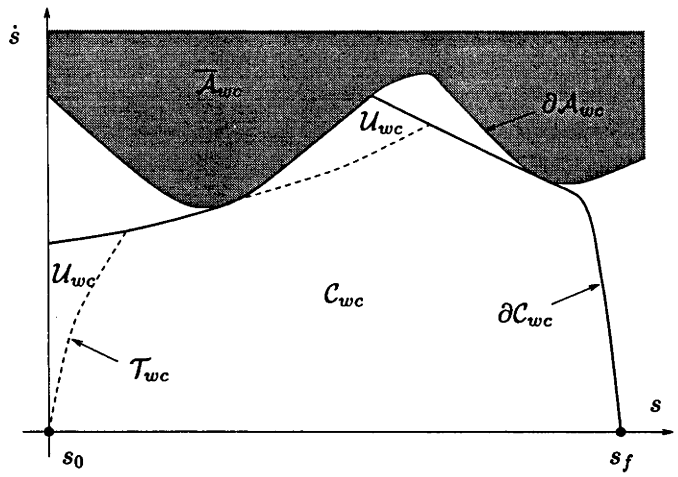


Figure 5.2: Worst-Case Admissible $\mathcal{A}_{wc}$, Controllable $\mathcal{C}_{wc}$ and Unreachable $\mathcal{U}_{wc}$ Regions, and the Worst-Case Time-Optimal Trajectory $\mathcal{T}_{wc}$.

The boundary $\partial\mathcal{A}_{wc}$ to $\mathcal{A}_{wc}$ is determined by the equation $\overline{\ddot{s}}_{min}(s,\dot{s}) - \underline{\ddot{s}}_{max}(s,\dot{s}) = 0$. Note that in Figure 5.2 $\mathcal{A}_{wc}$ is inferred by its complement $\overline{\mathcal{A}}_{wc}$.

The boundary $\partial\mathcal{C}_{wc}$ to $\mathcal{C}_{wc}$ can be determined by the following procedure.

1. Starting from the goal state $(s_f, 0)$, integrate backwards in time using $\ddot{s} = \underline{\ddot{s}}_{max}(s,\dot{s})$ until $s < s_0$, or until intersection with $\partial\mathcal{A}_{wc}$.

2. If intersection with $\partial\mathcal{A}_{wc}$ occurs, then track $\partial\mathcal{A}_{wc}$ backwards in s checking at each point whether a backward integration step using $\ddot{s} = \underline{\ddot{s}}_{max}(s,\dot{s})$ achieves a state $(s,\dot{s}) \in \mathcal{A}_{wc}$.

   - If it does, then leave $\partial\mathcal{A}_{wc}$ and integrate backwards in time as before.

3. Continue until $s < s_0$

Figure 5.2 illustrates a typical case that does not include *inadmissible islands* [60]. The region $\mathcal{C}_{wc}$ is bounded above by the curve $\partial\mathcal{C}_{wc}$ determined using the procedure above. Assuming that there are no inadmissible islands, then the area under $\partial\mathcal{C}_{wc}$ will be worst-case controllable. The proof of this follows from the observation that all states

above $\partial\mathcal{C}_{wc}$ have to use $\ddot{s} < \underline{\ddot{s}}_{max}(s, \dot{s})$, at least some of the time, to reach the goal state $(s_f, 0)^T$.

Let $\mathcal{T}_{wc}$ denote the *worst-case minimum time trajectory.* By worst-case minimum time trajectory we mean the time-optimal trajectory going from $(s_0, 0)^T$ to $(s_f, 0)^T$ while respecting the worst-case path acceleration constraints (5.6). This trajectory can be found using shooting methods or dynamic programming and will be the same as the *robust* time-optimal trajectory generated by the method in Chapter 4.

The regions such as $\mathcal{U}_{wc}$ which lie between $\mathcal{T}_{wc}$ and $\partial\mathcal{C}_{wc}$ are worst-case unreachable but are also worst-case controllable. The proposed scheme seeks to reduce cycle times without sacrificing robustness by taking advantage of the admissible range of $\ddot{s}$ that develops on-line to enter regions $\mathcal{U}_{wc}$ which are worst case controllable, but unreachable under worst case assumptions.

5.2.3 Proposed Feedback Control Law

Recall that the final time is inversely proportional to the area under the phase plane trajectory ($t_f = \int_0^{t_f} dt = \int_{s_0}^{s_f} \frac{1}{\dot{s}(s)} ds$). The fastest trajectory that is worst case controllable will be the one that most closely approximates $\partial\mathcal{C}_{wc}$ while remaining in $\mathcal{C}_{wc}$. This gives rise to the following feedback control law:

$$\ddot{s} = sat(\ddot{\sigma}; \underline{\ddot{s}}, \overline{\ddot{s}}) \ . \tag{5.8}$$

Here, $\ddot{\sigma}$ is the value of the path acceleration $\ddot{s}$ needed to intersect $\partial\mathcal{C}_{wc}$ at the end of the sample period, and $\underline{\ddot{s}}$ and $\overline{\ddot{s}}$ are the on-line limits on $\ddot{s}$ computed from (5.4) and (5.5).

At every sample period this feedback law utilises the full range of on-line admissible $\ddot{s}$ to seek intersection with $\partial\mathcal{C}_{wc}$ at the end of the sample period. If $\partial\mathcal{C}_{wc}$ can be reached, then the feedback law (5.8) chooses $\ddot{s} = \ddot{\sigma}$ to track it. Under worst-case assumptions (5.6), the trajectory will follow $\mathcal{T}_{wc}$, but it is more likely that the on-line admissible $\ddot{s}$ will allow entry into regions $\mathcal{U}_{wc}$ which are unreachable under worst-case assumptions.

Implementation requires the off-line determination of the curve $\partial\mathcal{C}_{wc}$ and its on-line storage as a numerical function of the path position s. The scheme requires a little more on-line computation than standard open-loop trajectory generation, but the memory required for the on-line storage of $\partial\mathcal{C}_{wc}$ is approximately only one-third of that required for the on-line storage of $\mathbf{q}_r(t)$, $\dot{\mathbf{q}}_r(t)$ and $\ddot{\mathbf{q}}_r(t)$.

5.3 Application of the Theory

In Chapter 4, we proposed a method for predicting the torque levels and controller gains required to reject to a specified level the errors which are excited by the plant-model disturbances during the time-optimal control of a manipulator. This method was based on the strategy of identifying the disturbance $\ddot{\mathbf{q}}_d(t)$ for a trajectory close to the "time-optimal" trajectory being sought, and then using it to choose the controller gains and the compensation torques to hold in reserve.

This formed the basis of the procedure described in § 4.2.2 which began with an experiment that identified $\ddot{\mathbf{q}}_d(t)$ using the "theoretical" time-optimal trajectory as reference input, and then used the results to choose more appropriate controller gains, compensation torques, and a reference trajectory for a second experiment. If the performance of the second experiment was not satisfactory, $\ddot{\mathbf{q}}_d(t)$ was identified from it, and the process continued.

We also use this procedure here, but we amend it to take account of the proposed closed-loop theory in § 5.2. The new procedure differs only in steps 7 and 9. In step 7, we compute the boundary $\partial\mathcal{C}_{wc}$ to the worst-case controllable region $\mathcal{C}_{wc}$ instead of computing an open-loop trajectory. In step 9, we run the closed-loop trajectory generator to generate the reference input to the closed-loop tracking controller, as opposed to providing the open-loop trajectory directly.

The Amended Identification Procedure

- We assume that we are given the path in the joint space, parameterised as a

function of the path parameter s viz $\mathbf{q}_r = \mathbf{f}(s)$, the nominal robot dynamics, torque limits of the form

$$\underline{\tau}_i(\mathbf{q}, \dot{\mathbf{q}}) \leq \tau_i \leq \overline{\tau}_i(\mathbf{q}, \dot{\mathbf{q}}) \tag{5.9}$$

and the error tolerance ϵ.

1. Calculate the time-optimal joint trajectory $\mathbf{q}_r(t)$ using the torque limits (5.9) - using, for example, the shooting method of Bobrow *et al.* [8], or the dynamic programming method described in Chapter 2.

2. Choose reasonable values for the experimental system gains $\mathbf{k}_p$. (We choose critical damping, i.e. $\mathbf{k}_v = 2\sqrt{\mathbf{k}_p}$.)

3. Drive the experimental system using the time-optimal joint trajectory $\mathbf{q}_r(t)$ as reference.

4. Calculate $\ddot{\mathbf{q}}_d(t)$ based on the output of this experiment.

5. Search for the gains $\mathbf{k}_p$ which produce $\|\hat{\mathbf{e}}\|_\infty = \epsilon$ when integrating $\ddot{\mathbf{q}}_d(t)$ through the error system

$$\ddot{\hat{\mathbf{e}}}(t) + \mathbf{k}_v\dot{\hat{\mathbf{e}}}(t) + \mathbf{k}_p\hat{\mathbf{e}}(t) = \ddot{\mathbf{q}}_d(t) \ , \tag{5.10}$$

and then calculate the associated compensation torques as a function of the path position, via

$$\boldsymbol{\tau}_{pd}(s) = \hat{\mathbf{M}}(\mathbf{q}_r(s)) \left\{\mathbf{k}_v\dot{\hat{\mathbf{e}}}(s) + \mathbf{k}_p\hat{\mathbf{e}}(s)\right\} \ . \tag{5.11}$$

6. Place an envelope around $\boldsymbol{\tau}_{pd}(s)$ and mirror in the zero-axis to yield $\underline{\boldsymbol{\tau}}_{pd}(s)$ and $\overline{\boldsymbol{\tau}}_{pd}(s)$ such that $\underline{\boldsymbol{\tau}}_{pd}(s) = -\overline{\boldsymbol{\tau}}_{pd}(s)$.

7. Calculate the boundary $\partial\mathcal{C}_{wc}$ to the worst-case controllable region $\mathcal{C}_{wc}$ using the algorithm described in § 5.2.2, with $\underline{\ddot{s}}_{max}(s, \dot{s})$ defined in Appendix 5.B and calculated using $\underline{\boldsymbol{\tau}}_{pd}(s)$ and $\overline{\boldsymbol{\tau}}_{pd}(s)$.

8. Set the experimental system gains $\mathbf{k}_p$ to be those which yielded $\|\hat{\mathbf{e}}\|_\infty = \epsilon$ in 5.

9. Run the closed-loop system - the closed-loop trajectory generator starts from an initial state $(s_0, \dot{s}_0)$ corresponding to the desired starting position and velocity (usually zero) and integrates $\ddot{s}$ provided by the feedback law described in § 5.2.3.

   This feedback law maximises the acceleration in attempting to drive/keep the trajectory to/on the worst-case controllable region boundary $\partial\mathcal{C}_{wc}$ without causing clipping of the commanded torques.

Note that if the online disturbances are greater than those predicted by the initial experiment, then it is possible that $\underline{\ddot{s}} > \overline{\ddot{s}}$. In this case, we have chosen to implement the control law $\ddot{s} = \frac{1}{2}(\underline{\ddot{s}} + \overline{\ddot{s}})$.

5.4 Experimental System and Performance Measures

The proposed closed-loop architecture was implemented on the first two links of our experimental manipulator, the 4 degree-of-freedom SCARA arm described in Appendix A. The model parameters used are those which were re-identified after the backlash was removed (see § 4.7 in Chapter 4), and are defined in Table 4.12 in Chapter 4. The torque limits were not affected by the backlash and are defined in equation (3.8) of Chapter 3.

We use the same performance measures of maximum error, torque utilisation and torque clipping as we did for the examples in Chapter 4, with the torque utilisation and clipping measures defined in equations (4.6), (4.7) and (4.8).

5.5 Example

In this section, we present some experimental results that illustrate the application of the method to one path tracking task. In the next section we present results from 3 randomly chosen paths by which we wish to demonstrate the robustness of the method.

To illustrate the new method, and in order to compare it with the previous robust trajectory planning method in Chapter 4, we consider the path tracking task that we

investigated in § 4.5; to track the joint space path

$$\mathbf{q}(s) = \begin{pmatrix} q_1(s) \\ q_2(s) \end{pmatrix} = \begin{pmatrix} \frac{\pi}{2}\sin(s) \\ \frac{\pi}{2}\cos(\frac{3}{2}s) \end{pmatrix}, \quad s \in [0, 2\pi]$$

in minimum time and with joint errors less than $\epsilon = (0.005, 0.0025)^T \; rad.$

The initial time-optimal trajectory was planned based on the model dynamics and assuming the actual torque bounds, using the trajectory planning method of Bobrow *et al.* [8]. This trajectory was then applied as the reference input to the closed-loop system without the closed-loop trajectory generator in place, using $\mathbf{k}_p = (100, 100)^T$ for the initial values of the controller gains. This yielded the errors shown in Figure 5.3. The tracking is well outside the prescribed tolerance, and the plot of the normalised command torque reveals that one or more of the commanded torques is being clipped for a large proportion of the time, Figure 5.4.

The acceleration disturbance $\ddot{\mathbf{q}}_d(t)$ was calculated from the output of this initial closed-loop experiment and is displayed as a function of the path position s in Figure 5.5.

The error system (5.10) was integrated with $\ddot{\mathbf{q}}_d(t)$ as input and with differing gains until the estimate of the errors $\hat{\mathbf{e}}(t)$ satisfied the prescribed tolerance. The gains which achieved this were $\mathbf{k}_p = (381.2, 865.6)^T$.

The compensation torque $\boldsymbol{\tau}_{pd}(s)$ was then estimated as a function of the path position s using equation (5.11), and the bounding functions $\underline{\boldsymbol{\tau}}_{pd}(s)$ and $\overline{\boldsymbol{\tau}}_{pd}(s)$ determined as shown in Figure 5.6.

The results produced to this point are very similar to those presented in § 4.5 of the previous chapter, *c.f.* Figures 4.3, 4.4, 4.5 and 4.6. This is not surprising because the only significant difference between the two cases is that the model parameters used here are those which were re-identified after the backlash was removed. However, it is at this point that the two methods diverge.

Instead of planning a time-optimal trajectory, we next planned the boundary $\partial\mathcal{C}_{wc}$ to the worst-case controllable region $\mathcal{C}_{wc}$ using the method described in § 5.2.2 with the torques backed off by $\underline{\boldsymbol{\tau}}_{pd}(s)$ and $\overline{\boldsymbol{\tau}}_{pd}(s)$. This is shown as the dot-dash line in

Figure 5.7.

Finally, the closed-loop system was activated. The closed-loop trajectory generator started from the initial state $(s_0, \dot{s}_0) = (0, 0)$, and integrated the path acceleration $\ddot{s}$ provided by the feedback law (5.8).

Figures 5.7 to 5.11 show plots of the results from the closed-loop scheme along with similar results obtained by applying the joint trajectory corresponding to the worst-case open-loop trajectory $\mathcal{T}_{wc}$ as reference to the closed-loop system without the closed-loop trajectory generator. ($\mathcal{T}_{wc}$ is defined using the shooting method of Bobrow *et al.* [8] with the same compensation torque bounds used to define $\partial\mathcal{C}_{wc}$.)

The phase plane plot in Figure 5.7 displays the boundary $\partial\mathcal{C}_{wc}$ to the worst-case controllable region $\mathcal{C}_{wc}$, the worst-case open-loop trajectory $\mathcal{T}_{wc}$, and the trajectory resulting from the closed-loop scheme. The plot shows that path velocity $\dot{s}$ of the trajectory resulting from the closed-loop scheme was everywhere greater than or equal to that of the equivalent trajectory resulting from the open-loop scheme, as was expected. The closed-loop scheme resulted in an elapsed time of $t_f = 9.553s$. This is significantly less than the elapsed time for the equivalent worst-case open-loop trajectory, $t_f = 10.130s$, and is extremely close to the nominal time-optimal tracking time of $t_f = 9.487s$, calculated using the model. The closed-loop trajectory always remained in the worst-case controllable region, as it was supposed to.

Figure 5.8, which displays the joint errors, reveals that the closed-loop system incorporating the closed-loop trajectory generator tracked the path within the specified tolerance, and that the errors for the closed-loop case, shown by the solid lines, were very similar to the errors produced by the equivalent open-loop example, shown by the dashed lines. Further, Figure 5.9 reveals that the actual compensation torques are very similar to those predicted (*c.f.* Figure 5.6), with the few minor violations of the bounds not being significant enough to cause a loss of tracking.

The average torque utilisation measure $\tau_{util} = 0.9798$ is very much higher for the closed-loop scheme than for the open-loop scheme, $\tau_{util} = 0.7104$, and very close to

the "true" time-optimal value $\tau_{util} = 1$. Indeed, Figure 5.10, which displays the torque utilisation and torque clipping for the closed-loop experiment, reveals that the actuators are fully utilised along much of the path, and that they are never oversaturated (*c.f.* Figure 5.11 which displays similar information for the equivalent open-loop example).

It would be reasonable to stop at this point, having achieved the desired tracking accuracy with an increase of less than 1% in the elapsed time and almost 100% torque utilisation. However, as in the examples in Chapter 4 we decided to iterate the method to see if better solutions were available, and to compare the convergence properties of the closed-loop scheme with those of the open-loop scheme.

Figures 5.12 to 5.15 summarise the results of 9 such iterations. Indeed there are marginally better solutions. However, the most noticeable thing is the consistency of the tracking times, tracking errors, torque utilisation and clipping and gains across the iterations. The convergence is to a much tighter region here, compared to the method of the previous chapter. Certainly, the evidence supporting the validity of our assumption that the disturbance processes acting for similar experiments will be similar is strengthened.

From this example it appears that the proposed method provides close to "true" time-optimal tracking after one iteration, so that further iterations yield only minimal improvement.

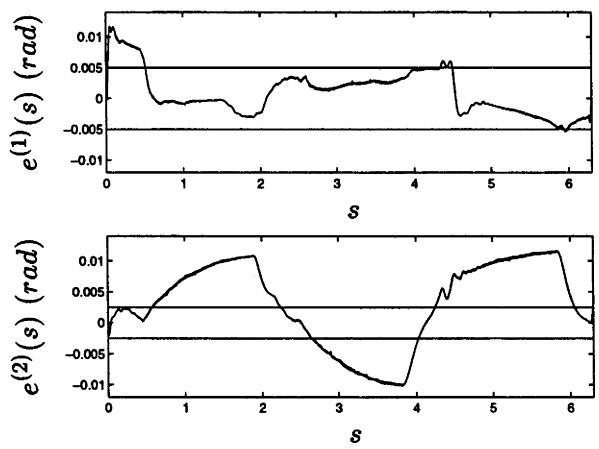


Figure 5.3: Tracking Errors - Initial Trajectory

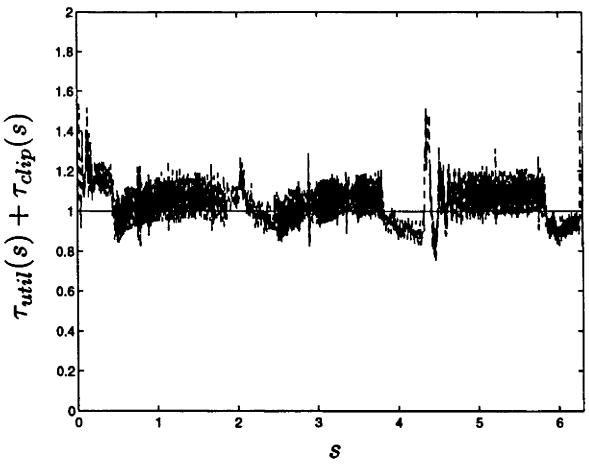


Figure 5.4: Normalised Command Torque - Initial Trajectory

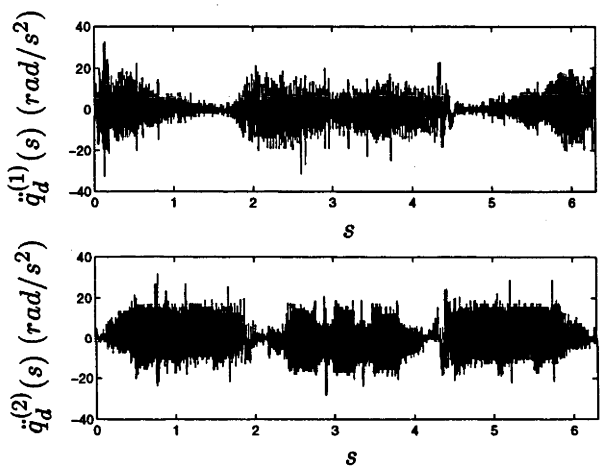


Figure 5.5: Acceleration Disturbance - Iteration 1

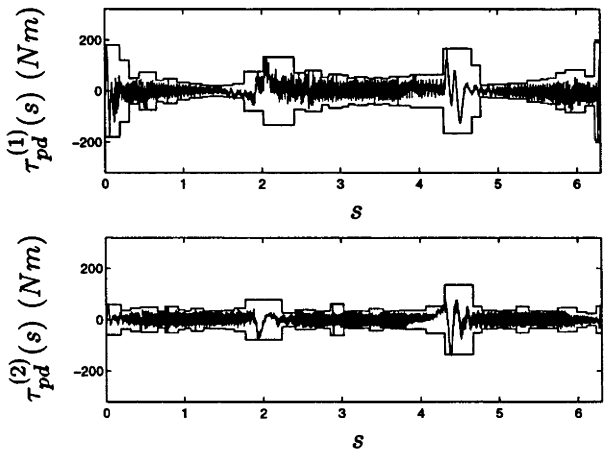


Figure 5.6: Predicted Compensation Torques and Bounds - Iteration 1

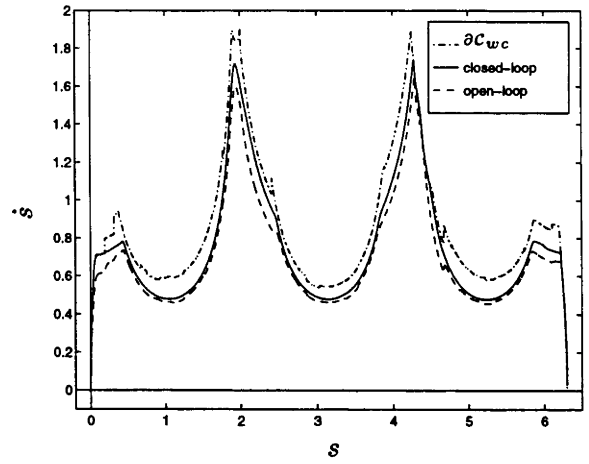


Figure 5.7: Phase Plane Trajectories

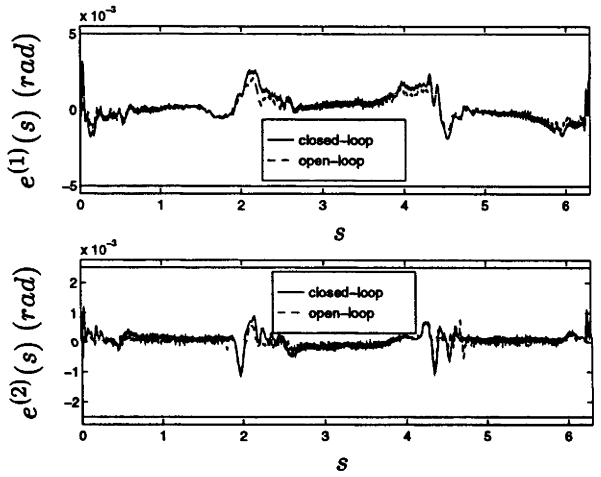


Figure 5.8: Closed-Loop and Open-Loop Tracking Errors - Iteration 1

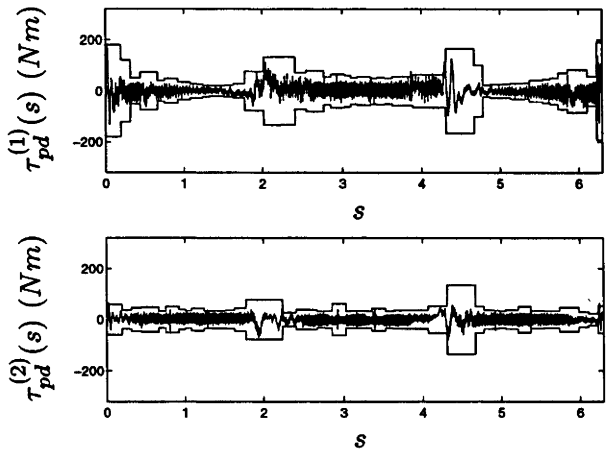


Figure 5.9: Actual Compensation Torques and Predicted Bounds - Iteration 1

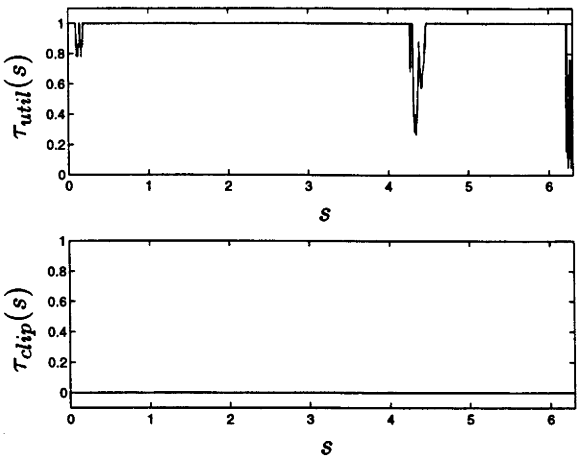


Figure 5.10: Closed-Loop Torque Utilisation and Clipping - Iteration 1

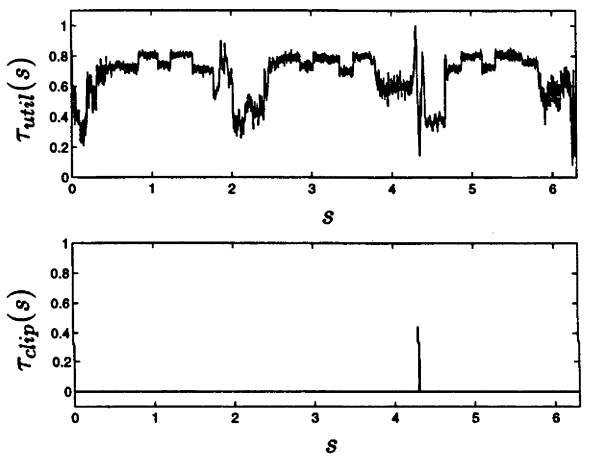


Figure 5.11: Open-Loop Torque Utilisation and Clipping - Iteration 1

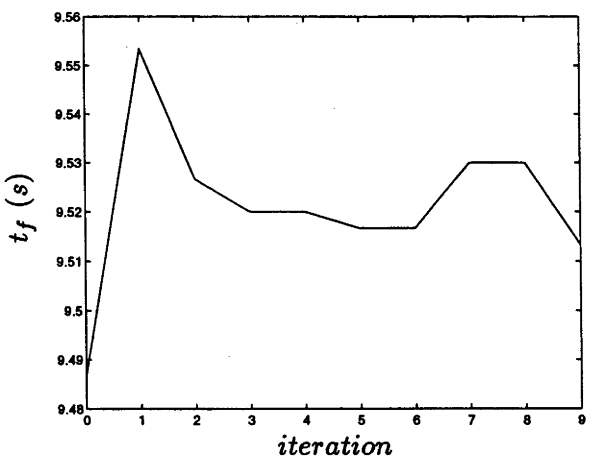


Figure 5.12: Tracking Time for 9 Iterations

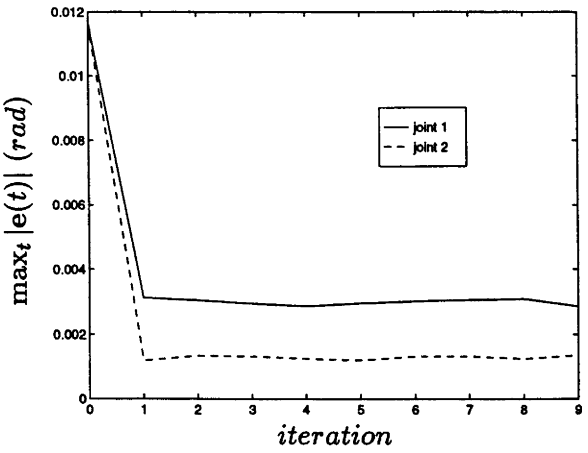


Figure 5.13: Maximum Error in Tracking For 9 Iterations

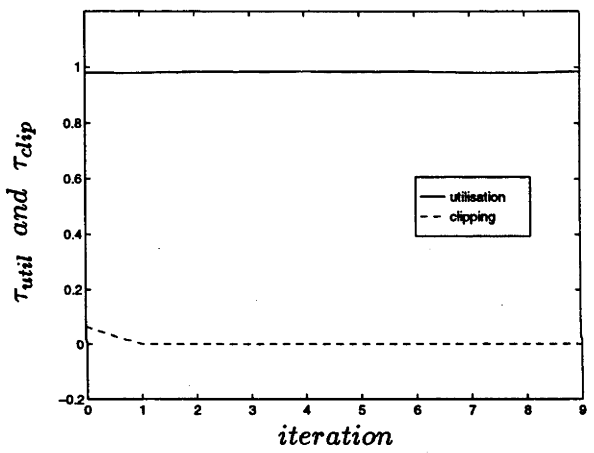


Figure 5.14: Torque Utilisation and Clipping For 9 Iterations

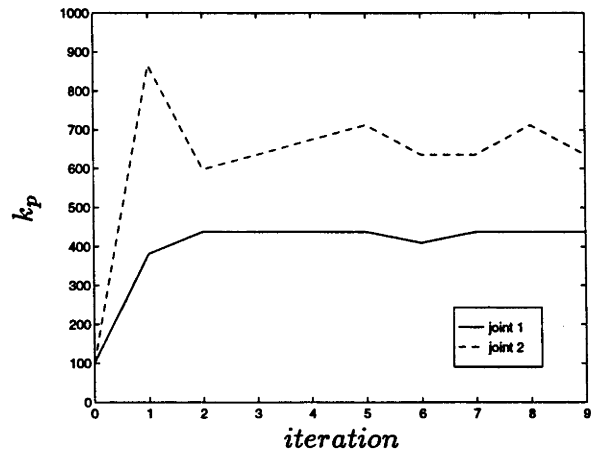


Figure 5.15: Proportional Gains For 9 Iterations

Iteration	Performance Measure						
	t_f (s)	$\max_t \|e(t)\|$ (rad)		τ_{util}	τ_{clip}	$\mathbf{k}_p$	
0	9.487	0.0117	0.0116	0.9798	0.0669	100.0	100.0
1	9.553	0.0031	0.0012	0.9791	0.0000	381.2	865.6
2	9.527	0.0030	0.0013	0.9838	0.0000	437.5	597.7
3	9.520	0.0029	0.0013	0.9843	0.0000	437.5	635.9
4	9.520	0.0029	0.0012	0.9848	0.0000	437.5	674.2
5	9.517	0.0030	0.0012	0.9845	0.0000	437.5	712.5
6	9.517	0.0030	0.0013	0.9851	0.0000	409.4	635.9
7	9.530	0.0031	0.0013	0.9804	0.0000	437.5	635.9
8	9.530	0.0031	0.0012	0.9799	0.0000	437.5	712.5
9	9.513	0.0029	0.0014	0.9845	0.0000	437.5	635.9

Table 5.1: Closed-Loop Trajectory Generation Results - Example Path

5.6 Robustness of the Method

The results in § 5.5 were excellent. However, they do not guarantee that the method will work as well in all cases. We wish to test the robustness of the method over more than this one task.

Ideally, we would have liked to have run this process for all 10 of the random paths that we used in the previous chapter. However, and as discussed at the end of Chapter 4, such an act might well provoke the problem of backlash, or indeed other effects, and so we choose not to do so. Instead, we select 3 of the random paths with which to demonstrate the robustness of the method.

The random paths selected are path 2 which looks relatively demanding, and paths 6 and 10 which were both affected by the backlash in the experiments of the last chapter and which we revisited there using the new model parameters. The full description of these paths is given in Appendix 4.A, and the (post-fix) time-optimal trajectory planning results are listed in Tables 4.14, 4.15, and 4.16.

As for each of the corresponding experiments in Chapter 4, we chose the tracking tolerance to be $\epsilon = (0.005, 0.005)^T$ rad, and for the initial closed-loop experiment for each path we set the values of the controller gains to be $\mathbf{k}_p = (100, 100)^T$. We iterated

the procedure 9 times for each path. The results of all of our experiments are detailed in Tables 5.2, 5.3 and 5.4 below.

Iteration	Performance Measure						
	t_f (s)	$\max_t \|e(t)\|$ (rad)		τ_{util}	τ_{clip}	k_p	
0	8.910	0.0105	0.0182	0.9903	0.0977	100.0	100.0
1	9.380	0.0043	0.0027	0.8912	0.0000	550.0	712.5

Table 5.2: Closed-Loop Trajectory Generation Results - Random Path 2

Iteration	Performance Measure						
	t_f (s)	$\max_t \|e(t)\|$ (rad)		τ_{util}	τ_{clip}	k_p	
0	8.903	0.0150	0.0106	0.9706	0.0580	100.0	100.0
1	9.147	0.0032	0.0033	0.9297	0.0000	409.4	291.4
2	9.227	0.0035	0.0030	0.9115	0.0000	437.5	291.4
3	9.073	0.0033	0.0032	0.9358	0.0000	465.6	253.1
4	9.030	0.0032	0.0035	0.9513	0.0000	465.6	234.0
5	9.123	0.0032	0.0036	0.9341	0.0000	465.6	234.0
6	9.153	0.0032	0.0034	0.9288	0.0000	465.6	234.0
7	9.230	0.0032	0.0037	0.9169	0.0000	465.6	234.0
8	9.133	0.0033	0.0032	0.9235	0.0000	465.6	253.1
9	9.383	0.0032	0.0037	0.8950	0.0000	465.6	234.0

Table 5.3: Closed-Loop Trajectory Generation Results - Random Path 6

Iteration	Performance Measure						
	t_f (s)	$\max_t \|e(t)\|$ (rad)		τ_{util}	τ_{clip}	k_p	
0	9.197	0.0130	0.0108	0.9631	0.0533	100.0	100.0
1	9.440	0.0035	0.0020	0.9298	0.0000	325.0	1095.3
2	9.413	0.0030	0.0041	0.9239	0.0000	325.0	310.5
3	9.347	0.0029	0.0023	0.9379	0.0000	353.1	942.2
4	9.290	0.0032	0.0042	0.9521	0.0000	325.0	291.4
5	9.393	0.0028	0.0023	0.9223	0.0000	353.1	942.2
6	9.263	0.0031	0.0035	0.9633	0.0000	325.0	406.2
7	9.310	0.0031	0.0024	0.9507	0.0000	339.1	865.6
8	9.330	0.0037	0.0034	0.9381	0.0000	325.0	406.2
9	9.380	0.0029	0.0024	0.9248	0.0000	353.1	789.1

Table 5.4: Closed-Loop Trajectory Generation Results - Random Path 10

Path 2 provides an interesting result. We stated earlier that not all tasks would be achievable. This would result from the tracking tolerance being set too small in

relation to the disturbances, providing compensation torques which were larger than those available. That is what happens here. We are able to iterate the process once, but the disturbance processes resulting from this experiment are sufficiently different from the initial experiment (using the nominal trajectory) that the compensation torques defined are larger than the available torques at certain configurations. This renders a second iteration impossible.

Even so, the result of the first iteration is better than any of the results provided by the previous method, *c.f.* Table 4.14, with the prescribed error tolerance being satisfied, an elapsed time of $t_f = 9.380s$ compared to a best of $t_f = 9.443s$ provided by the previous method, and an average torque utilisation measure of $\tau_{util} = 0.8912$ compared to $\tau_{util} = 0.7785$.

Paths 6 and 10 are not affected in this way, and the results that we obtain are excellent. The schemes provide robust tracking in one iteration, with the tracking errors meeting the prescribed tolerance thereafter, and remaining relatively constant. The elapsed times resulting from the first iteration show an increase of less than 3% from those for the nominal trajectories, and are much less than those for the previous method. The average torque utilisation measure suggests that the actuators are much more fully utilised.

In these two cases, the advantage in further iterating the process is more pronounced than for the example in § 5.5, although it is still very small. For path 6, the elapsed time can be reduced to $t_f = 9.030$ (iteration 5), compared to $t_f = 9.147$ after one iteration. For path 10, the elapsed time can be reduced to $t_f = 9.263$ (iteration 6), compared to $t_f = 9.440$ after the first iteration.

Tables 5.5, 5.6 and 5.7 compare typical results for the nominal trajectories and, "best"results for each of the two methods, for paths 2, 6 and 10 respectively. Here, best is selected as the fastest trajectory which satifies the tracking tolerance. Note that we were limited to iterating the process only once for path 2.

From these examples, it appears that the proposed method works well to achieve

almost "true" time-optimal tracking after one iteration, with some, although limited improvement to be gained by iterating the process further. This method is superior to that presented in Chapter 4.

Experiment	Performance Measure			
	t_f (s)	$\max_t \|e(t)\|$ (rad)	τ_{util}	τ_{clip}
Nominal	8.910	≈ 0.01	≈ 0.02	≈ 0.99	≈ 0.10
Open-Loop	9.443	0.0043	0.0046	0.7785	0.0004
Closed-Loop	9.380	0.0043	0.0027	0.8912	0.0000

Table 5.5: Comparison of Methods - Random Path 2

Experiment	Performance Measure				
	t_f (s)	$\max_t \|e(t)\|$ (rad)	τ_{util}	τ_{clip}
Nominal	8.903	≈ 0.015	≈ 0.01	≈ 0.97	≈ 0.06
Open-Loop	9.467	0.0047	0.0028	0.7226	0.0001
Closed-Loop	9.030	0.0032	0.0035	0.9513	0.0000

Table 5.6: Comparison of Methods - Random Path 6

Experiment	Performance Measure				
	t_f (s)	$\max_t \|e(t)\|$ (rad)	τ_{util}	τ_{clip}
Nominal	9.197	≈ 0.01	≈ 0.01	≈ 0.96	≈ 0.06
Open-Loop	9.847	0.0040	0.0050	0.6801	0.0001
Closed-Loop	9.263	0.0031	0.0035	0.9633	0.0000

Table 5.7: Comparison of Methods - Random Path 10

5.7 Conclusion

We have proposed a scheme for the closed-loop generation of trajectories for robust time-optimal robotic path tracking. The experimental results are consistent with the theory and demonstrate that tracking times are reduced compared to our previous approach in Chapter 4, while tracking accuracy and robustness remain approximately the same.

Appendices

5.A Online Admissible $\ddot{s}$

Differentiation of $\mathbf{q}_r(t) = \mathbf{f}(s(t))$ implies that $\dot{\mathbf{q}}_r(t) = \mathbf{f}'(s)\dot{s}$ and $\ddot{\mathbf{q}}_r(t) = \mathbf{f}'(s)\ddot{s} + \mathbf{f}''(s)\dot{s}^2$.

Substitution into the computed-torque control law (3.3) yields

$$\boldsymbol{\tau} = \mathbf{a}(s, \mathbf{q})\ddot{s} + \mathbf{b}(s, \dot{s}, \mathbf{q}, \dot{\mathbf{q}}) + \boldsymbol{\tau}_{pd} \,, \tag{5.12}$$

where

$$\mathbf{a}(s, \mathbf{q}) = \hat{\mathbf{M}}(\mathbf{q})\mathbf{f}'(s)$$

and

$$\mathbf{b}(s, \dot{s}, \mathbf{q}, \dot{\mathbf{q}}) = \hat{\mathbf{M}}(\mathbf{q})\mathbf{f}''(s)\dot{s}^2 + \hat{\mathbf{n}}(\mathbf{q}, \dot{\mathbf{q}}) \,.$$

Substitution of the expressions (5.12) for $\tau^{(i)}$ into the torque constraints (5.1) yields the following system of constraints

$$\underline{\tau}^{(i)}(\mathbf{q}, \dot{\mathbf{q}}) \leq a^{(i)}(s, \mathbf{q})\ddot{s} + b^{(i)}(s, \dot{s}, \mathbf{q}, \dot{\mathbf{q}}) + \tau_{pd}^{(i)} \leq \overline{\tau}^{(i)}(\mathbf{q}, \dot{\mathbf{q}}) \,. \tag{5.13}$$

The inequalities (5.13) for which $a^{(i)} \neq 0$ imply constraints on $\ddot{s}$, viz

$$\underline{\ddot{s}}^{(i)}(s, \dot{s}, \mathbf{q}, \dot{\mathbf{q}}) \leq \ddot{s} \leq \overline{\ddot{s}}^{(i)}(s, \dot{s}, \mathbf{q}, \dot{\mathbf{q}}) \,,$$

where

$$\underline{\ddot{s}}^{(i)} = \begin{cases} (\underline{\tau}^{(i)}(\mathbf{q}, \dot{\mathbf{q}}) - b^{(i)}(s, \dot{s}, \mathbf{q}, \dot{\mathbf{q}}) - \tau_{pd}^{(i)})/a^{(i)}(s, \mathbf{q}) & \text{if } a^{(i)}(s, \mathbf{q}) > 0 \\ (\overline{\tau}^{(i)}(\mathbf{q}, \dot{\mathbf{q}}) - b^{(i)}(s, \dot{s}, \mathbf{q}, \dot{\mathbf{q}}) - \tau_{pd}^{(i)})/a^{(i)}(s, \mathbf{q}) & \text{if } a^{(i)}(s, \mathbf{q}) < 0 \end{cases} \,, \tag{5.14}$$

and

$$\overline{\ddot{s}}^{(i)} = \begin{cases} (\overline{\tau}^{(i)}(\mathbf{q}, \dot{\mathbf{q}}) - b^{(i)}(s, \dot{s}, \mathbf{q}, \dot{\mathbf{q}}) - \tau_{pd}^{(i)})/a^{(i)}(s, \mathbf{q}) & \text{if } a^{(i)}(s, \mathbf{q}) > 0 \\ (\underline{\tau}^{(i)}(\mathbf{q}, \dot{\mathbf{q}}) - b^{(i)}(s, \dot{s}, \mathbf{q}, \dot{\mathbf{q}}) - \tau_{pd}^{(i)})/a^{(i)}(s, \mathbf{q}) & \text{if } a^{(i)}(s, \mathbf{q}) < 0 \end{cases} \,. \tag{5.15}$$

The simultaneous satisfaction of these constraints is equivalent to

$$\underline{\ddot{s}}(s, \dot{s}, \mathbf{q}, \dot{\mathbf{q}}) \leq \ddot{s} \leq \overline{\ddot{s}}(s, \dot{s}, \mathbf{q}, \dot{\mathbf{q}}) \,,$$

where

$$\underline{\ddot{s}} = \max_{\{i:a^{(i)}(s,\mathbf{q})\neq 0\}} \underline{\ddot{s}}^{(i)} ,$$

and

$$\overline{\ddot{s}} = \min_{\{i:a^{(i)}(s,\mathbf{q})\neq 0\}} \overline{\ddot{s}}^{(i)} .$$

5.B Offline Bounds on Admissible $\ddot{s}$

If we substitute $\mathbf{q}(t) \approx \mathbf{q}_r(t) = \mathbf{f}(s(t))$ and its derivatives into the computed torque law (3.3), then we obtain

$$\boldsymbol{\tau} = \mathbf{a}(s,\dot{s})\ddot{s} + \mathbf{b}(s,\dot{s}) + \boldsymbol{\tau}_{pd} , \tag{5.16}$$

where

$$\mathbf{a}(s) = \hat{\mathbf{M}}(\mathbf{f}(s))\mathbf{f}'(s)$$

and

$$\mathbf{b}(s,\dot{s}) = \hat{\mathbf{M}}(\mathbf{f}(s))\mathbf{f}''(s)\dot{s}^2 + \hat{\mathbf{n}}(\mathbf{f}(s),\dot{\mathbf{f}}'(s)\dot{s}) .$$

The torque constraints can be written as functions of s and $\dot{s}$ viz

$$\underline{\boldsymbol{\tau}}(s,\dot{s}) \leq \boldsymbol{\tau} \leq \overline{\boldsymbol{\tau}}(s,\dot{s}) , \tag{5.17}$$

where

$$\underline{\boldsymbol{\tau}}(s,\dot{s}) = \underline{\boldsymbol{\tau}}(\mathbf{f}(s),\mathbf{f}'(s)\dot{s}) ,$$

and

$$\overline{\boldsymbol{\tau}}(s,\dot{s}) = \overline{\boldsymbol{\tau}}(\mathbf{f}(s),\mathbf{f}'(s)\dot{s}) .$$

Substitution of the expressions (5.17) for $\tau^{(i)}$ into the torque constraints (5.16) yields the following system of constraints

$$\underline{\tau}^{(i)}(s,\dot{s}) \leq a^{(i)}(s)\ddot{s} + b^{(i)}(s,\dot{s}) + \tau_{pd}^{(i)} \leq \overline{\tau}^{(i)}(s,\dot{s}) \tag{5.18}$$

The inequalities (5.18) have the same form as those in (5.13) and lead to equations for $\underline{\ddot{s}}^{(i)}$ and $\overline{\ddot{s}}^{(i)}$ in the same form as (5.14) and (5.15).

By substituting the constraints (5.3) on $\tau_{pd}^{(i)}$ into (5.18) and rearranging, we obtain the following limits on $\underline{\ddot{s}}^{(i)}$ and $\overline{\ddot{s}}^{(i)}$:

$$(\underline{\ddot{s}}^{(i)})_{min} \leq \underline{\ddot{s}} \leq (\underline{\ddot{s}}^{(i)})_{max}$$

and

$$(\overline{\ddot{s}}^{(i)})_{min} \leq \overline{\ddot{s}} \leq (\overline{\ddot{s}}^{(i)})_{max}$$

where

$$(\underline{\ddot{s}}^{(i)})_{min} = \begin{cases} (\underline{\tau}^{(i)} - b^{(i)}(s,\dot{s}) - \overline{\tau}_{pd}^{(i)})/a^{(i)}(s) & \text{if } a^{(i)}(s) > 0 \\ (\overline{\tau}^{(i)} - b^{(i)}(s,\dot{s}) - \underline{\tau}_{pd}^{(i)})/a^{(i)}(s) & \text{if } a^{(i)}(s) < 0 \end{cases},$$

$$(\underline{\ddot{s}}^{(i)})_{max} = \begin{cases} (\underline{\tau}^{(i)} - b^{(i)}(s,\dot{s}) - \underline{\tau}_{pd}^{(i)})/a^{(i)}(s) & \text{if } a^{(i)}(s) > 0 \\ (\overline{\tau}^{(i)} - b^{(i)}(s,\dot{s}) - \overline{\tau}_{pd}^{(i)})/a^{(i)}(s) & \text{if } a^{(i)}(s) < 0 \end{cases},$$

$$(\overline{\ddot{s}}^{(i)})_{min} = \begin{cases} (\overline{\tau}^{(i)} - b^{(i)}(s,\dot{s}) - \overline{\tau}_{pd}^{(i)})/a^{(i)}(s) & \text{if } a^{(i)}(s) > 0 \\ (\underline{\tau}^{(i)} - b^{(i)}(s,\dot{s}) - \underline{\tau}_{pd}^{(i)})/a^{(i)}(s) & \text{if } a^{(i)}(s) < 0 \end{cases},$$

and

$$(\overline{\ddot{s}}^{(i)})_{max} = \begin{cases} (\overline{\tau}^{(i)} - b^{(i)}(s,\dot{s}) - \underline{\tau}_{pd}^{(i)})/a^{(i)}(s) & \text{if } a^{(i)}(s) > 0 \\ (\underline{\tau}^{(i)} - b^{(i)}(s,\dot{s}) - \overline{\tau}_{pd}^{(i)})/a^{(i)}(s) & \text{if } a^{(i)}(s) < 0 \end{cases}.$$

The simultaneous satisfaction of these constraints is equivalent to

$$\underline{\ddot{s}}_{min}(s,\dot{s}) \leq \underline{\ddot{s}} \leq \underline{\ddot{s}}_{max}(s,\dot{s})$$

and

$$\overline{\ddot{s}}_{min}(s,\dot{s}) \leq \overline{\ddot{s}} \leq \overline{\ddot{s}}_{max}(s,\dot{s})$$

where

$$\underline{\ddot{s}}_{min} = \max_{\{i:a^{(i)}(s)\neq 0\}} (\underline{\ddot{s}}^{(i)})_{min},$$

$$\underline{\ddot{s}}_{max} = \min_{\{i:a^{(i)}(s)\neq 0\}} (\underline{\ddot{s}}^{(i)})_{max},$$

$$\overline{\ddot{s}}_{min} = \max_{\{i:a^{(i)}(s)\neq 0\}} (\overline{\ddot{s}}^{(i)})_{min},$$

and

$$\overline{\ddot{s}}_{max} = \min_{\{i:a^{(i)}(s)\neq 0\}} (\overline{\ddot{s}}^{(i)})_{max}.$$

Chapter 6

Fast Pick and Place at Kinematic Singularities

Abstract

In this chapter, we investigate whether singular configurations may be good sites for high-speed pick and place operations. Motivation comes from the observation that the end-effector velocity can be brought to zero at a singularity without stopping the mechanism. This allows the robot to keep some of its kinetic energy while accomplishing the pick or place task and can lead to faster cycle times compared to pick and place at nearby regular configurations.

6.1 Introduction

Singular configurations of nonredundant serial manipulators have been studied extensively (see, for example, [37] and references therein), but few authors have suggested any practical uses for them. Hunt has cited their ability to withstand large loads in certain directions [35] and it is easy to see that humans exploit the mechanical advantage offered by singularities in tasks such as standing, walking, weightlifting and archery [38].

The main point of the work in this chapter is to suggest a new way in which singular configurations might be exploited for practical benefit. Our proposition is

that kinematic singularities may be good sites for high-speed pick and place operations because they have the potential to yield faster cycle times compared to sites at nearby regular configurations. The key to achieving this result is to exploit the manipulator's ability to stop its end-effector at the singularity without stopping the motion of its links. This allows the robot to perform the pick or place operation without a complete loss of kinetic energy.

In § 6.2 we present a simple example that illustrates this potential for cycle time reduction. In § 6.3 we discuss the problem and its constraints more generally. In § 6.4 we address how we might achieve the pick or place task, and we split this into two distinct parts. In § 6.4.1 we derive a constraint on the joint path kinematics that ensures that the end-effector becomes stationary at the singularity regardless of the path timing, and derive additional kinematic constraints which define the path geometries at and near the singular configuration which are likely to result in the end-effector remaining in the proximity of the pick and place site for longer. However, these kinematic constraints do not consider the issue of timing and so we cannot use them to predict the amount of time that the end-effector dwells at and/or near the pick and place site. Thus, in § 6.4.2 we convert these kinematic path constraints into dynamic constraints on the path timing in an attempt to ensure that the end-effector stays within a prescribed neighbourhood of the pick and place site for a chosen amount of time. In § 6.5 we apply these constraints to the example. In § 6.6 we discuss some practical considerations and finally, in § 6.7 we draw conclusions.

6.2 Illustrative Example

In this section, we present an example based on an industrial SCARA robot that we have in our laboratory. In addition to illustrating the effect of interest, we attempt to quantify the potential for reducing cycle times by comparing time-optimal trajectories that use a singularity as a pick and place site to those that use nearby regular configuration sites.

To make the problem more tractable, we constrain the motion of the end-effector to track a given path in a plane. This reduces the number of states in the dynamic optimisation problem from 4 (for our planar example), the joint displacements and velocities, to 2, the path displacement and velocity. However, it should be noted that the problem can be solved without recourse to this, as will be discussed in § 6.3.

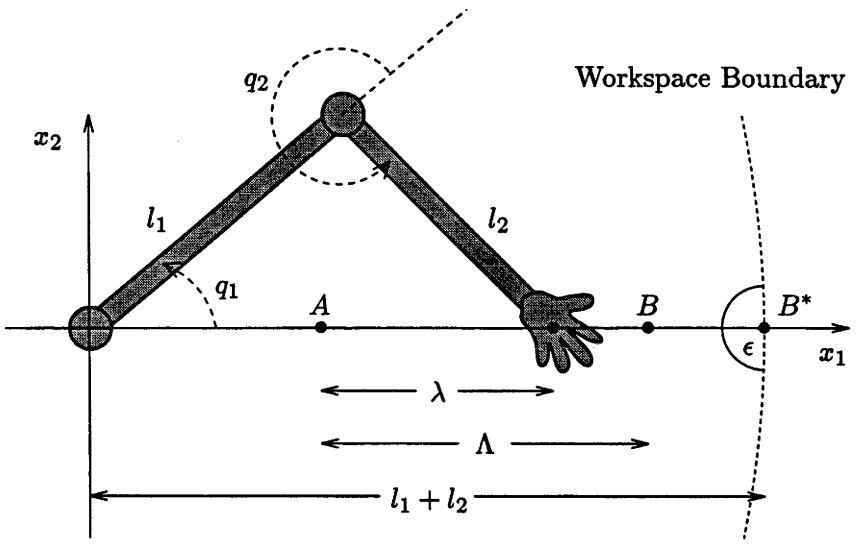


Figure 6.1: 2-DOF SCARA Manipulator and Straight Line Path

Consider the two-link manipulator shown in Figure 6.1. The task is to move the end point on a straight line from the point A to a point B and then back to A in minimum time. The robot must start and finish at rest, and it must bring the tip velocity to zero at B in order to perform the pick or place operation.

We assume that the coordinates of A are given and that B can be anywhere on a radial line outward from A to B^*, which is on the workspace boundary. Our objective is to compare time-optimal cycle times for all possible sites of B, including B^* which corresponds to an outstretched singular configuration of the robot.

From the Jacobian equation,

$$\dot{\mathbf{x}} = \mathbf{J}(\mathbf{q})\dot{\mathbf{q}},$$

which maps the joint velocities $\dot{\mathbf{q}} \in \mathbb{R}^2$ onto the end point velocities $\dot{\mathbf{x}} \in \mathbb{R}^2$, it is clear

that if $\mathbf{J}(\mathbf{q})$ is nonsingular then $\dot{\mathbf{q}}$ must be zero to achieve $\dot{\mathbf{x}} = \mathbf{0}$. This means that the robot must come to a complete stop at all sites B $\neq$ B*. (At B^*, $\mathbf{J}(\mathbf{q})$ is singular.) As a result, time optimal trajectories for these cases can be determined as the sum of two independent time-optimal trajectories: $A \rightarrow B$ and $B \rightarrow A$, each starting and ending at rest.

To make the example realistic we use the dynamic model described in Appendix A which corresponds to the SCARA manipulator in our laboratory. This has the form

$$\hat{\mathbf{M}}(\mathbf{q})\ddot{\mathbf{q}} + \hat{\mathbf{d}}(\mathbf{q}, \dot{\mathbf{q}}) + \hat{\mathbf{v}}(\dot{\mathbf{q}}) = \boldsymbol{\tau} \tag{6.1}$$

where $\hat{\mathbf{M}}(\mathbf{q})$, $\hat{\mathbf{d}}(\mathbf{q}, \dot{\mathbf{q}})$ and $\hat{\mathbf{v}}(\dot{\mathbf{q}})$ represent the mass-matrix, coriolis-centrifugal and friction vectors respectively, defined in equations (A.2), (A.3) and (A.4).

The parameter values for $\hat{\mathbf{M}}(\mathbf{q})$, $\hat{\mathbf{d}}(\mathbf{q}, \dot{\mathbf{q}})$ and $\hat{\mathbf{v}}(\dot{\mathbf{q}})$, identified using the standard least squares technique, are shown in Table 6.1. Note that in this example, the parameters of the cosine terms in the centrifugal-coriolis vector were not available and are set to zero.

The torque limits for the robot are given by

$$\left.\begin{array}{rcl} \underline{\tau}(\dot{\mathbf{q}}) & = & \max \ \{-206.28, -1910.0 - 984.987\dot{\mathbf{q}}\} \\ \overline{\tau}(\dot{\mathbf{q}}) & = & \min \ \{+206.28, +1910.0 - 984.987\dot{\mathbf{q}}\} \end{array}\right\}, \tag{6.2}$$

and are due to current and voltage limits of $4A$ and $100V$ respectively.

We position A at $\mathbf{x} = ((l_1 + l_2)/2, 0)^T$ and calculate our results based on a shooting method similar to that described by Bobrow *et al.* in [8]. Figure 6.2 plots the time optimal $A \rightarrow B \rightarrow A$ cycle times for sites B, other than B^*, versus the distance, Λ, between A and B. Confirming intuition, the curve demonstrates that the cycle time increases with increasing distance between points A and B.

Now let us consider motion to $B = B^*$. The joint path

$$\mathbf{q}(s) = \begin{pmatrix} q_1(s) \\ q_2(s) \end{pmatrix} = \begin{pmatrix} s \\ sin^{-1}(-\frac{l_1}{l_2} sin(s)) - s \end{pmatrix}, \tag{6.3}$$

$s \in [-1.0041, 1.0041]$, causes the end point to trace the straight lines $A \rightarrow B^*$ and $B^* \rightarrow A$. The robot begins in an elbow-down configuration with $\mathbf{x} = A$, passes through

Parameter	Value	Units	Represents
f_{11}	-0.1718	Nms^2/rad	Inertia terms
$f_{12} = f_{21}$	-0.0859	Nms^2/rad	
g_{11}	-6.3674	Nms^2/rad	
$g_{12} = g_{21}$	-3.1837	Nms^2/rad	
h_{11}	43.0900	Nms^2/rad	
$h_{12} = h_{21}$	0.1973	Nms^2/rad	
h_{22}	21.5020	Nms^2/rad	
v_{1s}	6.3674	Nms^2/rad^2	Centrifugal
v_{2s}	-3.1837	Nms^2/rad^2	and coriolis
w_{1s}	-3.1837	Nms^2/rad^2	terms
v_{1c}	0.0000	Nms^2/rad^2	
v_{2c}	0.0000	Nms^2/rad^2	
w_{1c}	0.0000	Nms^2/rad^2	
f_{1p}	15.3007	Nm	Friction terms
f_{1n}	-14.8315	Nm	
f_{2p}	14.0112	Nm	
f_{2n}	-14.5492	Nm	
b_{1p}	32.0320	Nms/rad	
b_{1n}	30.5636	Nms/rad	
b_{2p}	19.2845	Nms/rad	
b_{2n}	21.8976	Nms/rad	
l_1	0.6100	m	Length of links
l_2	0.5800	m	1 and 2

Table 6.1: SCARA Manipulator Parameter Data

the singular configuration at $\mathbf{x} = B^*$, and finishes in an elbow-up configuration with $\mathbf{x} = A$ once more.

The kinematics for this example are simply

$$\mathbf{x} = \mathbf{f}(\mathbf{q})) = \begin{pmatrix} l_1 \cos(q_1) + l_2 \cos(q_1 + q_2) \\ l_1 \sin(q_1) + l_2 \sin(q_1 + q_2) \end{pmatrix}, \tag{6.4}$$

and with $\mathbf{q} = \mathbf{q}(s)$, the Jacobian equation describing the velocity of the end-effector is

$$\dot{\mathbf{x}} = \mathbf{J}(\mathbf{q}(s))\mathbf{q}'(s)\dot{s}, \tag{6.5}$$

where the Jacobian $\mathbf{J}(\mathbf{q}(s))$ is

$$\mathbf{J}(\mathbf{q}(s)) = \begin{pmatrix} -l_1 \sin(q_1(s)) - l_2 \sin(q_1(s) + q_2(s)) & -l_2 \sin(q_1(s) + q_2(s)) \\ l_1 \cos(q_1(s)) + l_2 \cos(q_1(s) + q_2(s)) & l_2 \cos(q_1(s) + q_2(s)) \end{pmatrix}, \tag{6.6}$$

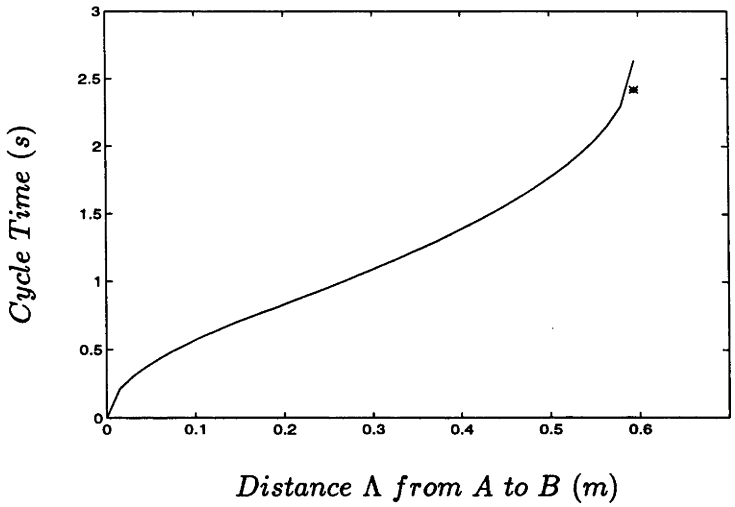


Figure 6.2: Time t versus Distance Λ

and the derivative of the path (6.3) is given by

$$\mathbf{q}'(s) = \begin{pmatrix} 1 \\ \frac{-l_1 \cos(s)}{\sqrt{l_2^2 - l_1^2 \sin^2(s)}} - 1 \end{pmatrix} . \tag{6.7}$$

At the singular configuration $\mathbf{q} = \mathbf{0} \equiv s \triangleq s^* = 0$, and

$$\mathbf{J}(\mathbf{q}(s^*)) = \begin{pmatrix} 0 & 0 \\ l_1 + l_2 & l_2 \end{pmatrix} , \tag{6.8}$$

which is singular, as expected, and

$$\mathbf{q}'(s^*) = \begin{pmatrix} 1 \\ -\frac{l_1}{l_2} - 1 \end{pmatrix} . \tag{6.9}$$

It is clear to see that $\mathbf{J}(\mathbf{q}(s^*))\mathbf{q}'(s^*) = \mathbf{0}$. Hence, at the singularity, $\mathbf{q}'(s) \neq \mathbf{0}$ is in the null space of $\mathbf{J}(\mathbf{q}(s))$, and $\dot{\mathbf{x}} = \mathbf{J}(\mathbf{q}(s))\mathbf{q}'(s)\dot{s}$ provides $\dot{\mathbf{x}} = \mathbf{0}$ at B^* regardless of $\dot{s}$. That is, the velocity of the end-effector is zero at the singularity no matter how fast the path is traversed.

The time-optimal trajectory for this joint path is computed using the same SCARA dynamics, torque limits and shooting method as before.

Figure 6.2 confirms that the resulting cycle time for B^*, $2.420s$, is some 8% faster than the time-optimal cycle time of its nearest neighbouring regular configuration site, $2.634s$, and is equal to the cycle time of a site B which is 1.5% closer to A than B^*.

Further insight can be obtained from Figure 6.3 which compares the $s - \dot{s}$ phase plane plots for two time-optimal trajectories that use site B^*. (Note that the shooting method we employ is based on finding the fastest trajectory, or *maximum velocity curve*, respecting the robot dynamics (6.1) but constrained within a region of the $s - \dot{s}$ phase plane, the *admissible region* $\mathcal{A}$, whose boundaries are a function of the dynamics (6.1), the torque limits (6.2) and the description of the path (6.3).)

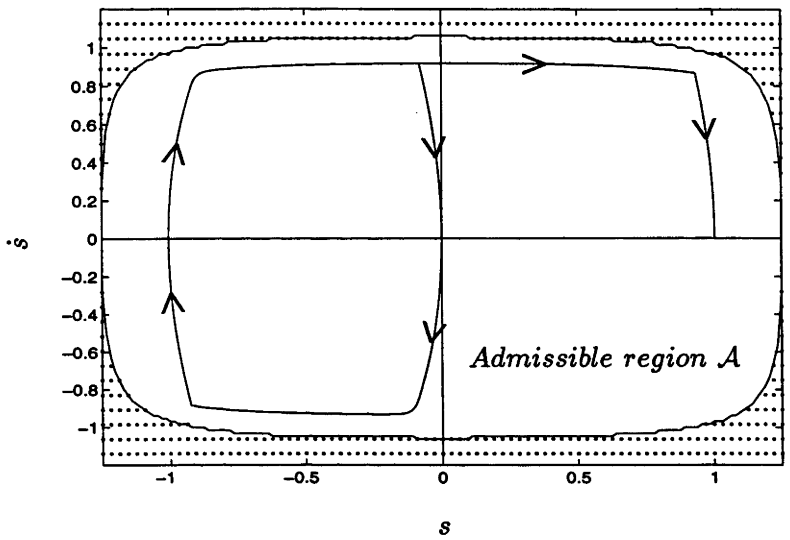


Figure 6.3: Phase Plane Trajectories

The first, going directly from $s = -1.0041$ to $s = 1.0041$, corresponds to the trajectory which does not stop at the singularity. The second plot, going from $s = -1.0041$ to $s = 0$ and then back to $s = -1.0041$, results from adding a constraint that the manipulator must stop completely at B^* (when $s = 0$) as if B^* were a regular site. Because the cycle time is inversely proportional to the area under the phase plane trajectory (recall that

$$t_f = \int_0^{t_f} dt = \int_{s_0}^{s_f} \frac{dt}{ds} ds = \int_{s_0}^{s_f} \frac{1}{\dot{s}(s)} ds \ ),$$

it is clear that the time saving results from the elimination of this constraint. The manipulator keeps its kinetic energy while the end-effector stops at B^*.

The time saving for this particular example is modest, but the effect is nonethe-

less apparent. More dramatic time savings would result if the trajectory had longer acceleration and deceleration phases. The acceleration phases were very short for our example because the motors very quickly reached their velocity limit (where the back EMF equals the 100 volt limit of the power supply). Doubling the voltage limit for this example results in Figures 6.4 and 6.5 which show a more significant time saving of 22%, from $1.954s$ to $1.517s$ in the two cases. The cycle time equals that of a site B which is 15% closer to A than is B^*.

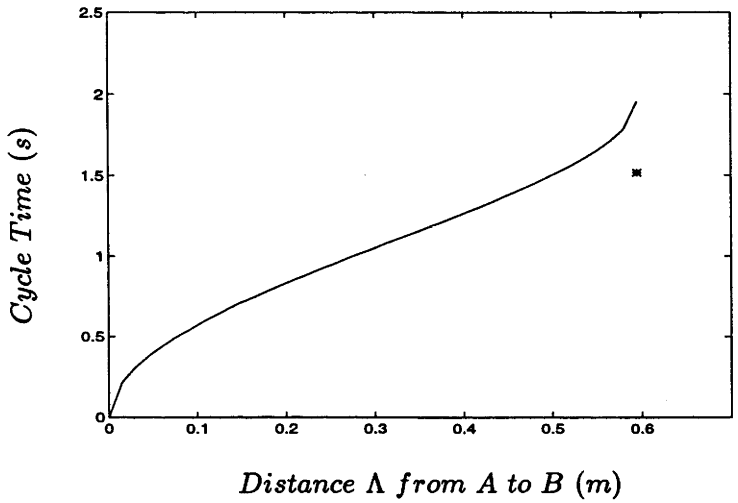


Figure 6.4: Time t versus Distance Λ - Higher Voltage Limit

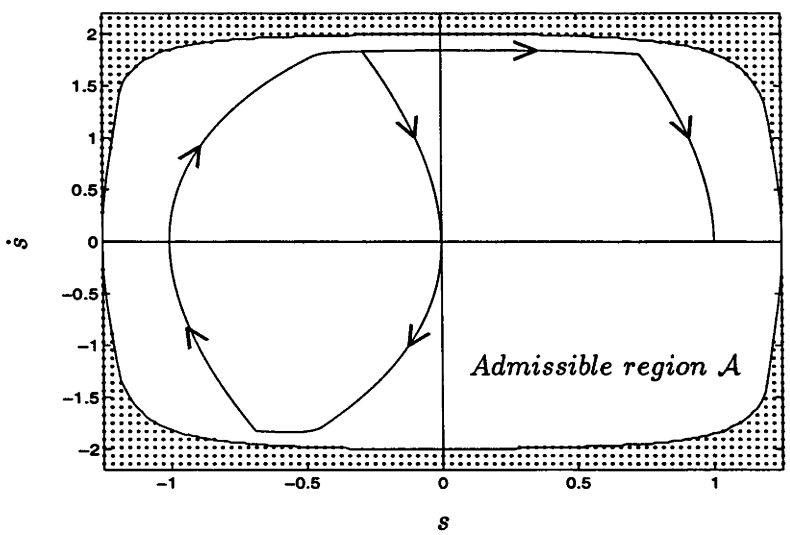


Figure 6.5: Phase Plane Trajectories - Higher Voltage Limit

6.3 The Real Problem

The examples above highlight a potential for using singularities to reduce cycle times for pick and place operations. However, this potential is not fully exploited because the end-effector is artificially constrained to a path. In general, this will not be the case, and the problem will be one of *point-to-point* control where the only constraint on the path of the end-effector will be that it passes through the points A and B or B^*, *c.f.* Figure 6.6. (Of course, there are more complex variations of this, for example where more than two points are concerned, but all such cases can be broken up into smaller yet similar subproblems.) Furthermore, it is not clear that the robot's end-effector will remain stationary at the singular site long enough to perform the pick or place task.

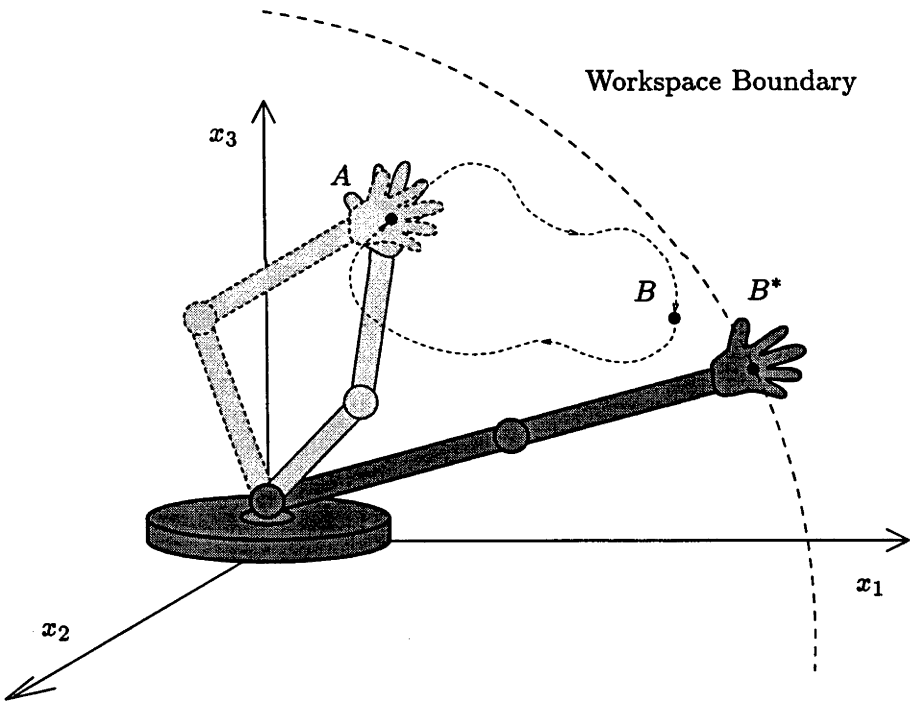


Figure 6.6: The Generalised Case

To determine the true time-optimal trajectory for a pick or place operation at a singular site, the joint path constraint must be removed and the full state (joint positions and velocities) dynamic optimisation problem must be solved. In addition to the usual stationary boundary conditions at the initial and final configurations, the

solution trajectory $\mathbf{q}(t)$ has to satisfy the following intermediate constraint which brings the end-effector velocity to zero at the pick and place site:

- Constraint 1: $\mathbf{q}(t)$ must pass through $\mathbf{q}^*$ with $\dot{\mathbf{q}} \in \text{null}\{\mathbf{J}(\mathbf{q}^*)\}$.

where $\mathbf{q}^*$ represents the singular joint configuration. Furthermore, to ensure adequate time and precision for the pick or place operation, the solution trajectory $\mathbf{q}(t)$ must also satisfy the constraint:

- Constraint 2: $\mathbf{x}(t)$ must remain in a neighbourhood, ϵ, of $\mathbf{x}^*$ for a given duration T.

where $\mathbf{x}^*$ represents the position of the singularity. Thus, the dynamic optimisation problem becomes a three-point boundary value problem plus the additional constraint 2.

Because the solution of this problem is somewhat involved and adds little physical insight, we choose not to make it a focus of this work. Instead, we will focus on the small motion kinematics of paths and trajectories in an ϵ-neighbourhood of $\mathbf{x}^*$. We show below that constraints 1 and 2 define the true time-optimal trajectory in the neighbourhood of $\mathbf{q}^*$ whenever constraint 2 is active. Thus, we can determine the trajectory in the neighbourhood of $\mathbf{q}^*$ without solving the dynamic optimisation problem.

In the usual case $\mathbf{J}(\mathbf{q}^*)$ has one degree of rank deficiency (assumed henceforth). Then constraint 1 can be interpreted as constraining $\mathbf{q}(t)$ to a path in the neighbourhood of $\mathbf{q}^*$. With this interpretation, constraint 2 limits the rate of path traversal in the neighbourhood of $\mathbf{q}^*$. Thus, if constraint 2 is active, then the two constraints together define the trajectory $\mathbf{q}(t)$ in a local sense in the neighbourhood of $\mathbf{q}^*$. This is a very useful result because it allows the full state optimisation problem described above to be split into two independent problems that have fixed initial and final boundary conditions and no additional constraints.

If constraint 2 is not active, then the path timing in the neighbourhood of $\mathbf{q}^*$ will depend on the dynamics. In this case, the optimisation problem described above can be solved as a three-point boundary value problem with constraint 2 excluded.

6.4 Achieving the Task at the Pick and Place Site

In the design of a strategy to achieve the task at the pick and place site, there are two questions that we wish to answer:

- What path geometry is appropriate at or near the singular configuration for retaining joint motion whilst stopping the end-effector motion ?
- How do we re-time the manipulator at or near the singular configuration so the the end-effector stays at the pick and place site long enough to perform the given task ?

The first relates to the design of the path locally in the neighbourhood of the singularity and so is a question involving kinematics, and the second to the timing of the traversal of the path which is a question of dynamics.

6.4.1 Instantaneous Kinematics at the Singularity

In this section, we assume that constraint 2 is active and we seek to define an appropriate local joint path geometry satisfying both constraints 1 and 2.

To satisfy constraint 1, the joint path must be tangent to the null space of $\mathbf{J}$ at $\mathbf{q}^*$. However, there are infinitely many such paths, and some will lead to lower time-optimal cycle times than others. To account for this, we choose to define the local joint path based on the following assumption:

- The true time-optimal joint trajectory will locally minimise the ratio of the end-effector displacement to the joint velocity.

Such a trajectory will tend to maximise the robot's kinetic energy at $\mathbf{q}^*$ subject to constraint 2 which limits movement of the end-effector.

The following derivation defines differential constraints on the joint path that bring the end-effector velocity to zero at the singularity and tend to minimise the end-effector motion in the neighbourhood of the singularity no matter how fast the joint path is traversed.

Let $\mathbf{x}(t)$ represent the end-effector position as a function of time. Let t^* represent the time at the pick and place site. A Taylor series expansion of $\mathbf{x}(t^*)$ about t^* gives

$$\mathbf{x}(t^* + \Delta t) = \mathbf{x}(t^*) + \dot{\mathbf{x}}(t^*)\Delta t + \frac{1}{2}\ddot{\mathbf{x}}(t^*)\Delta t^2 + \frac{1}{6}\dddot{\mathbf{x}}(t^*)\Delta t^3 + \ldots \tag{6.10}$$

Now the end-effector will be stationary at $\mathbf{x}(t^*)$ if $\dot{\mathbf{x}}(t^*) = \mathbf{0}$. Furthermore, minimisation of higher order derivatives, in the order $\ddot{\mathbf{x}}(t^*), \dddot{\mathbf{x}}(t^*), \ldots$, will ensure that the end-effector remains in the neighbourhood of $\mathbf{x}(t^*)$ for a longer period.

Let $\mathbf{q}(s)$ represent the joint path and $\mathbf{q}(s(t))$ represent the timed joint path, or joint trajectory. Furthermore, let the joint path include a singular configuration $\mathbf{q}(t^*)$ at $s = s^*$ and let the pick and place site $\mathbf{x}(t^*)$ be the end-effector position corresponding to $\mathbf{q}(t^*)$.

Differentiation of the kinematic map $\mathbf{x}(t) = \mathbf{f}(\mathbf{q}(s(t)))$ using the chain rule provides the following expressions:

$$\dot{\mathbf{x}}(t^*) = \mathbf{J}\mathbf{q}'\dot{s} \;|_{s=s^*,t=t^*} \tag{6.11}$$

$$\ddot{\mathbf{x}}(t^*) = (\mathbf{J}\mathbf{q}')'\dot{s}^2 + \mathbf{J}\mathbf{q}'\ddot{s} \;|_{s=s^*,t=t^*} \tag{6.12}$$

$$\dddot{\mathbf{x}}(t^*) = (\mathbf{J}\mathbf{q}')''\dot{s}^3 + 3(\mathbf{J}\mathbf{q}')'\dot{s}\ddot{s} + \mathbf{J}\mathbf{q}'\dddot{s} \;|_{s=s^*,t=t^*} \tag{6.13}$$

Because $\mathbf{J} = \mathbf{J}(\mathbf{q})$ is singular we can choose

$$\mathbf{q}'(t^*) \in \text{null}\{\mathbf{J}\}. \tag{6.14}$$

This makes $\mathbf{J}\mathbf{q}' = \mathbf{0}$, ensuring that the last term on the right side of each of equations (6.11)-(6.13) is equal to zero regardless of the path timing, $s(t)$. Most importantly, it forces the end-effector velocity $\dot{\mathbf{x}}(t^*)$ to equal zero.

The end-effector acceleration, $\ddot{\mathbf{x}}(t^*)$, and jerk, $\dddot{\mathbf{x}}\ (t^*)$, can be minimised by choosing the path $\mathbf{q}(s)$ to minimise $(\mathbf{J}\mathbf{q}')'$ and $(\mathbf{J}\mathbf{q}')''$. Expansion of these terms

$$(\mathbf{J}\mathbf{q}')' = \mathbf{J}'\mathbf{q}' + \mathbf{J}\mathbf{q}'' \tag{6.15}$$

$$(\mathbf{J}\mathbf{q}')'' = \mathbf{J}''\mathbf{q}' + 2\mathbf{J}'\mathbf{q}'' + \mathbf{J}\mathbf{q}''' \tag{6.16}$$

reveal that in general there are no solutions for $\mathbf{q}''(t^*)$ and $\mathbf{q}'''(t^*)$ that can force $(\mathbf{J}\mathbf{q}')'$ and $(\mathbf{J}\mathbf{q}')''$ to zero because the rank of $\mathbf{J}$ is not full. However these terms can be minimised by choosing $\mathbf{q}''(t^*)$ and $\mathbf{q}'''(t^*)$ as follows:

$$\mathbf{q}''(t^*) = \mathbf{J}^+(-\mathbf{J}'\mathbf{q}')\ |_{s=s^*} \tag{6.17}$$

$$\mathbf{q}'''(t^*) = \mathbf{J}^+(-\mathbf{J}''\mathbf{q}' - 2\mathbf{J}'\mathbf{q}'')\ |_{s=s^*}\ . \tag{6.18}$$

Here, $\mathbf{J}^+$ represents the pseudo-inverse of $\mathbf{J}$.

For any given path timing $s(t)$, these solutions sequentially minimise $\dot{\mathbf{x}}(t^*)$, $\ddot{\mathbf{x}}(t^*)$, and $\dddot{\mathbf{x}}\ (t^*)$. These solutions also ensure that $(\mathbf{J}\mathbf{q}')'$ and $(\mathbf{J}\mathbf{q}')''$ lie in the the complement to the image space of $\mathbf{J}$, which means $\ddot{\mathbf{x}}(t^*)$ and $\dddot{\mathbf{x}}\ (t^*)$ will lie in the the complement to the image space of $\mathbf{J}\ |_{s=s^*}$, regardless of path timing.

Similar equations for $\mathbf{q}''''(t^*)$, $\mathbf{q}'''''(t^*)$, ..., can be derived in the same way. Each one ensures that another derivative of $\mathbf{x}(t)$ is minimised and lies in the complement to the image space of $\mathbf{J}$ evaluated at the singular configuration.

The geometric interpretation of this can be derived using the picture in Figure 6.7. By ensuring that $\dot{\mathbf{x}} = 0$ and sequentially minimising the higher order derivatives of $\mathbf{x}$, we are pushing the variation $\mathbf{x}(t^* + \Delta t) - \mathbf{x}(t^*)$ into the complement to the image space of $\mathbf{J}(\mathbf{q})$, *c.f.* equation (6.10). This can viewed as forcing the end-effector path to approach the pick and place site along increasingly higher order cusps, whose limiting case, when an infinite number of derivatives of $\mathbf{x}$ have been minimised, is the infinite order cusp colinear with the complement to the image space of $\mathbf{J}(\mathbf{q})$. Our example of § 6.2 is such a case, where by design $\mathbf{q}^{(n)}(s)\ |_{s=s^*}, i = 1, ..., \infty$ all belong to the complement to the image space of $\mathbf{J}(\mathbf{q}^*)$.

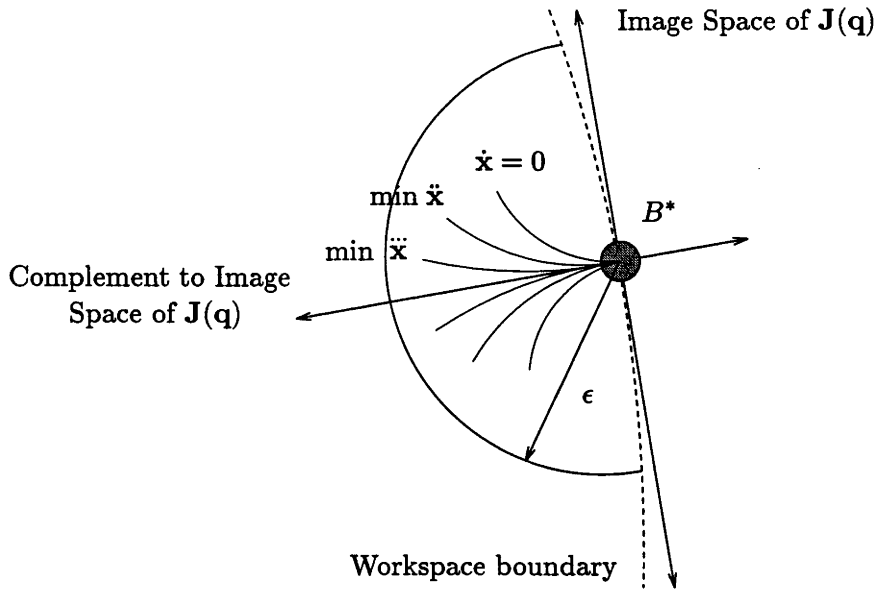


Figure 6.7: Local Path Kinematics - Sequential Minimisation of Derivatives of $\mathbf{x}(t^*)$

6.4.2 Local Timing Constraints

Although the path constraints developed in the previous section attempt to ensure that the end-effector velocity will tend to remain in the neighbourhood of the pick and place site, irrespective of the path timing, there is no guarantee that the end-effector will remain at that site long enough to perform the pick or place operation. In many applications the end-effector need only be stationary for a moment, e.g. the time it takes to open a pneumatic gripper. However, this moment must be guaranteed.

In this section, we derive constraints that attempt to ensure that the end-effector remains within a specified neighbourhood $\|\Delta\mathbf{x}\| < \epsilon$ of $\mathbf{x}(t^*)$ for a chosen time duration T (i.e. satisfies constraint 2).

Because we have chosen to make $\dot{\mathbf{x}}(t^*) = \mathbf{0}$, and assuming that $\Delta\mathbf{x}$ is small, the Taylor series expansion about $\mathbf{x}(t^*)$, (6.10), will be dominated by the quadratic term in Δt, and so

$$\Delta \mathbf{x} \approx \frac{1}{2}\ddot{\mathbf{x}}(t^*)\Delta t^2. \tag{6.19}$$

Using this term to approximate the series we find that

$$\|\Delta \mathbf{x}\| < \epsilon \quad \text{for} \quad \Delta t \in \left[-\sqrt{\frac{2\epsilon}{\|\ddot{\mathbf{x}}(t^*)\|}}, \sqrt{\frac{2\epsilon}{\|\ddot{\mathbf{x}}(t^*)\|}} \right].$$

Thus, if approximation (6.19) is valid, then

$$T \approx 2\sqrt{\frac{2\epsilon}{\|\ddot{\mathbf{x}}(t^*)\|}}$$

which implies that $\|\Delta \mathbf{x}\| < \epsilon$ for duration T if

$$\|\ddot{\mathbf{x}}(t^*)\| < \frac{8\epsilon}{T^2}. \tag{6.20}$$

We will now translate constraint (6.20) into an equivalent constraint on the path timing.

If the instantaneous joint path kinematics are chosen to respect the constraints derived in § 6.4.1, then equations (6.12)-(6.18) provide the following expressions

$$\ddot{\mathbf{x}}(t^*) = \boldsymbol{\alpha}\dot{s}^2(t^*) \tag{6.21}$$

$$\dddot{\mathbf{x}}\,(t^*) = \boldsymbol{\beta}\dot{s}^3(t^*) + 3\boldsymbol{\alpha}\dot{s}(t^*)\ddot{s}(t^*) \tag{6.22}$$

where

$$\boldsymbol{\alpha} = (\mathbf{I} - \mathbf{J}\mathbf{J}^+)\mathbf{J}'\mathbf{q}' \mid_{s=s^*}$$

$$\boldsymbol{\beta} = (\mathbf{I} - \mathbf{J}\mathbf{J}^+)(\mathbf{J}''\mathbf{q}' + 2\mathbf{J}'\mathbf{q}'') \mid_{s=s^*}.$$

Here $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are vectors that lie in the complement to the image space of $\mathbf{J}$.

Substitution of (6.21) into (6.20) provides the following constraint on $\dot{s}(t^*)$:

$$\dot{s}^2(t^*) < \frac{8\epsilon}{\|\boldsymbol{\alpha}\|T^2}. \tag{6.23}$$

This constraint attempts to ensure that $\|\Delta \mathbf{x}\| < \epsilon$ for a duration T. However, it is valid only for ϵ and T small enough to ensure that the approximation (6.19) is valid.

For cases when higher order terms in the Taylor series expansion are significant, it is possible to choose higher derivatives of $s(t)$ to eliminate them or lessen their magnitudes.

For example, if $\mathbf{J}$ has a rank deficiency of 1, then $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ will be colinear and the following choice of $\ddot{s}(t^*)$ will ensure that $\dddot{\mathbf{x}}\ (t^*) = \mathbf{0}$:

$$\ddot{s} = \frac{\dot{s}^2}{3\gamma} \text{ where } \boldsymbol{\alpha} = \gamma\boldsymbol{\beta}. \tag{6.24}$$

Otherwise $\ddot{s}$ can be chosen to minimise $\dddot{\mathbf{x}}\ (t^*)$ using the following equation which ensures orthogonality between $\ddot{\mathbf{x}}(t^*)$ and $\dddot{\mathbf{x}}\ (t^*)$:

$$\ddot{s} = -\boldsymbol{\alpha}^{+}\boldsymbol{\beta}(\frac{\dot{s}^2}{3}).$$

6.5 Example Revisited

We now show how the theory in § 6.4.2 can be applied to the examples in § 6.2 to ensure that the end-point remains at the singular site long enough to perform the task, in this case, to open the gripper.

From experimentation we have determined that the pneumatic gripper takes $\approx$ $100ms$ to open. We choose to ignore the timing delay since this is $\approx 1ms$. Thus we take $T = 100ms$. We choose $\epsilon = 3mm$ to correspond to the robot's repeatability.

Using these values and the robot kinematics at point B^*, inequality (6.23) provides the velocity constraint $\dot{s}(t^*) < 1.3848$ and equation (6.24) the acceleration constraint $\ddot{s} = 0$.

Figure 6.3 reveals that the unconstrained trajectory does not exceed this path velocity constraint. In fact, the end-effector stays within the $3mm$-neighbourhood for $\approx 150ms$ which is more than the $100ms$ required. Thus constraint 2 is inactive.

If we consider the case with an increased voltage limit, Figure 6.5, the unconstrained trajectory clearly exceeds the path velocity constraint. Using the same shooting method as before, we recompute the time-optimal trajectory with the path velocity

constraint, applied at $s = s^* = 0$, and the acceleration constraint, applied over the interval $s \in [-\dot{s}(t^*)\frac{T}{2}, \dot{s}(t^*)\frac{T}{2}] = [-0.0692, 0.0692]$. Figure 6.8 compares the unconstrained trajectory with the new constrained trajectory.

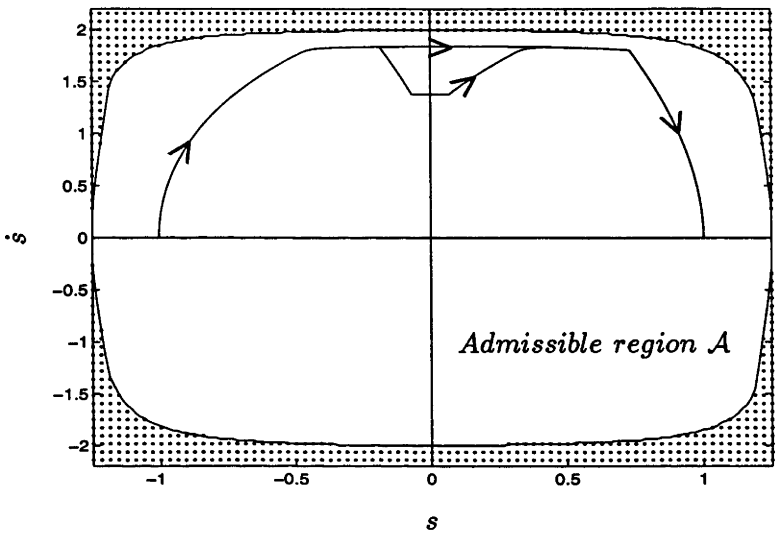


Figure 6.8: Phase Plane Trajectories

Postmortem analysis shows that without the constraints the end-effector remains within the requisite $3mm$ of B^* for only $75ms$. However, with the constraints, the end-effector stays within the requisite $3mm$ of B^* for $100ms$, as predicted. The cycle time increases from $1.517s$ to $1.574s$, but is still 19% faster than using the neighbouring regular site B, and is as fast as a regular site B that is 12% closer to A than is B^*.

6.6 Practical Considerations

Our approach to the problem has been theoretical, with subsequent testing of the theory through simulation. However, the theory does not consider some more practical aspects of the problem which would be apparent in a real set-up, and which would need to be accounted for in the design of any control strategy.

One such example is in the forces that will be exerted on an object by the end-effector as it moves towards gripping the object. (Remember that the end-effector is

moving in the vicinity of the pick and place site, stopping only instantaneously at the site itself.) The question might then be *How large can the contact force be so that the object is not damaged* ? Another similar consideration is what happens when the end-effector releases an object. In this case the object may have some velocity. The question might then be *What is the maximum velocity that the object can have so that it remains within the desired drop region* ? In both of these cases, the motion of the end-effector might introduce rotations in the object which might result in the object falling on release.

One obvious way to address the problem of keeping the end-effector at the pick and place site for long enough to perform the task would be to use a manipulator which possesses an extra degree-of-freedom. This degree of redundancy might then be utilised to allow the end-effector to dwell at the pick and place site for an arbitrarily long period. However, in this case, the pick and place site would not be located at a singular configuration, and an alternative formulation of the problem would need to be considered.

Our solution considers minimising higher order derivatives until the dwell time constraint is satisfied. The number of minimisations that we perform is subjective in the sense that once the constraint is met, continuing the minimisation procedure will not subsequently cause violation of the constraint. However, there might be good reason to continue the minimisation process if the optimality criteria is not purely minimum-time. For example if the energy expended or power dissipated in achieving the task is an issue, then there might be paths based on higher order minimisation which will yield a more conservative energy/power with little or no cost in time.

6.7 Conclusions

We have proposed a new idea that singular configurations may be good sites for pick and place operations because the end-effector velocity can be brought to zero without stopping the manipulator. We have shown that cycle times can be reduced compared

to using nearby regular configurations. Simulations suggest that the saving in time is modest for our SCARA industrial robot arm, but may be more dramatic for robots that have slower rates of acceleration relative to their maximum speed. Because the pick or place operation has to be done "on the fly", we have developed a method which guarantees that the end-effector remains within a given neighbourhood of the desired site for a given amount of time. Simulations show that this method is accurate.

Chapter 7

Conclusions

7.1 Conclusions

In this thesis, we have considered three problems relating to the time-optimal motion control of robot manipulators:

1. Trajectory Planning using Dynamic Programming.
2. Robust Time-Optimal Path Tracking Control.
3. Fast Pick and Place at Kinematic Singularities.

There follows a summary of the results and conclusions for each of these problems.

Trajectory Planning using Dynamic Programming

In this work, we investigated the use of dynamic programming to determine time-optimal trajectories for robotic path tracking applications. Our approach differed to previous dynamic programming approaches to this problem in that we solved numerically the continuous optimal-control problem specified by the *Hamilton-Jacobi-Bellman* partial differential equation of dynamic programming using finite difference Markov chain type approximations. Our desire was to evaluate the applicability of the resulting solutions to a real manipulator and to analyse issues relating to the convergence of this approach.

We conclude as follows:

- We have demonstrated that dynamic programming is computationally feasible for this problem.
- We have applied solutions of the dynamic programming algorithm to a real SCARA manipulator with good results - the manipulator tracks the solution trajectories closely and exhibits no undesirable behaviour.
- We have shown that the rate of convergence of the numerical scheme is consistent with that predicted by theory.
- We have found that the minimum-time end-point (target) constraint imposes severe limitation regarding the rate of convergence and computational speed of the dynamic programming algorithm (in particular, commonly used acceleration methods do not work).
- The PMP based shooting method is superior to dynamic programming when applied to the standard minimum-time criteria, and is the method of choice.
- The dynamic programming algorithm can easily be modified to handle more general optimisation criteria, in which case the advantages of the PMP approach diminish (because the optimal control is no longer bang-bang in general, and a two-point boundary value problem needs to be solved).

Robust Time-Optimal Path Tracking Control

In this work, our goal was to provide a systematic way of controlling industrial robots to achieve accurate and time-optimal tracking of specified paths. The major issue that we addressed was to take previously developed theory and to develop ways of making it practical, to apply to real industrial robots. In particular, our focus has been to address the issue of how a user specified tracking tolerance can be achieved in spite of unmodelled dynamics.

To this end, we have developed three major results:

- Based on representing modelling errors as disturbances in joint accelerations, we have developed a theory for relating these disturbances to the performance of robots under computed-torque control. The theory links the *acceleration disturbances* to the controller tuning, the tracking errors, and the *compensation torques.* Experimental results confirmed that the theory works well in practice.

- We have used this theory in a predictive way to provide a method for planning trajectories which are robust to modelling disturbances. The method involves identifying the disturbances using a trajectory close to the time-optimal trajectory being sought. The theory is then used to predict the compensation torques that need to be held on reserve during trajectory planning, and the controller gains that are required to reject the expected levels of disturbance to a specified tolerance. Experiments have shown that if trajectories are planned and the controllers tuned in this way, then the robot performs near time-optimal tracking of the path, accurate to the specified tracking tolerance.

- We have extended this method to a less conservative closed-loop architecture for on-line trajectory generation. We proposed a feedback law for setting the reference path acceleration such that path tracking is nearly time-optimal, robustly controllable, and accurate to a prescribed tolerance. Experimental results have demonstrated that the tracking times are reduced compared to the first approach while the tracking accuracy and robustness remain approximately the same.

Fast Pick and Place at Kinematic Singularities

In this work, we investigated a new idea; that singular configurations might be good sites with respect to reducing cycle times of pick and place operations.

- We have shown that cycle times of pick and place operations can be reduced by placing the pick and place sites at singular configurations, compared to nearby

regular configurations.

- We have developed a method which guarantees that the end-effector remains within a given neighbourhood of the pick and place site for a given amount of time - long enough to perform the pick or place task. Simulations show that this method is accurate.

7.2 Further Research

In this section, we discuss some open problems related to the work in this thesis. There are issues relating to the practical implementation of our schemes which might be viewed as open problems, for example the path is normally specified as a series of splines and not as a parameterised function, and tracking tolerances would normally be specified in the task space and not in the joint space. However, it is not in doubt that such problems are relatively straightforward to solve. In this respect, they do not constitute "open" problems for research.

Some open research problems that we have identified include:

- In our work on the dynamic programming solution to the optimal path timing problem, we found that acceleration methods commonly used to speed up the rate of convergence and computational speed of the dynamic programming algorithm did not work. This was due to the target constraint. Given that it is not unusual to have target states defined in optimal control problems, given the large overheads required to compute accurate solutions, and given that acceleration methods improve the rate of convergence by an order of magnitude, it would be of interest to investigate whether this could be overcome.

- Time-optimal solutions are not kind to robot manipulators. This was no more clearly demonstrated than in our experiments in Chapter 4 where repeated application of minimum-time trajectories caused substantial backlash to develop. Ways need to be considered of making time-optimal trajectories more kind to

robot manipulators.

- The joints of our manipulator are equipped with high resolution encoders (360,000 counts per revolution). However, it is not unusual for manipulators to be manufactured with encoders possessing a resolution an order of magnitude less than this (high resolution encoders are relatively expensive). In such cases, the levels of noise introduced by taking numerical derivatives of the joint measurements would be significantly increased, and there there would be no guarantee that the theory presented in Chapter 3 would predict accurately. Also, it is possible that we would have to implement filters when estimating the joint velocities for use on-line in order to prevent instabilities occuring. The theory takes no account of filters. One interesting question is *How much resolution is needed to circumvent the use of filters* ? Another, which is related more to economic considerations, is *Is there a case for designing robots with higher resolution encoders to achieve results using theories such as that developed here* ?

- The theory in Chapter 3 assumes rigid-body dynamics. In practice, this is not the case. For example, flexibilities may be present in harmonic drives on the actuators and/or in light links. In these cases, the end-effector position calculated through the kinematic model using the encoder measurements does not necessarily equate to the actual end-effector position and prediction of the tracking may be difficult. This is also true if backlash effects are present. It would be valuable to develop the theory to account for such effects.

- When considering if singular configurations might be good sites for pick and place operations, we recognised the need to slow down at the pick and place for long enough for the pick or place task to occur. However, there were other practical aspects that we did not consider, for example, the forces that would be exerted on an object when it is grasped or released. The validity of our approach was demonstrated in simulation. However, in order to apply the ideas to real manipulators successfully, it is likely that practical issues such as these would

first need to be addressed. (A more detailed discussion of these issues appears in § 6.6 of Chapter 6.)

Appendix A

Experimental System

The experimental work in Chapters 2 through 6 is performed or based on a 4 degree-of-freedom SCARA manipulator that we have in our laboratory, Figure A.1.

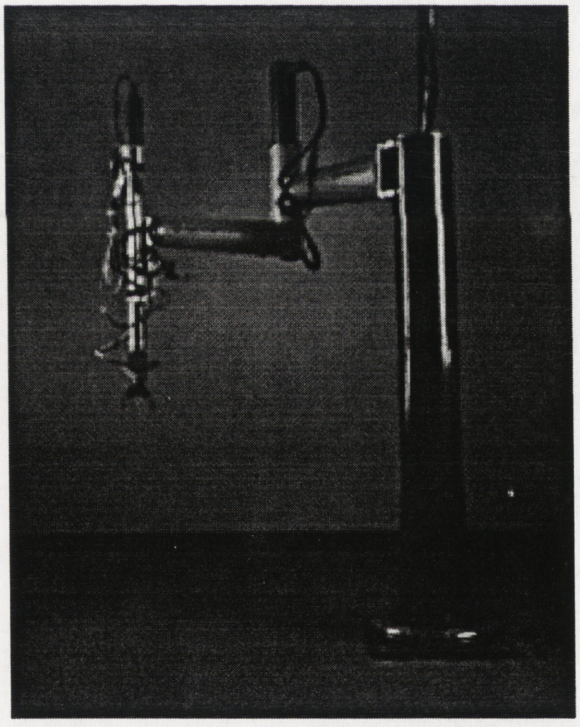


Figure A.1: 4 Degree-of-Freedom SCARA Robot

The joints are driven by current controlled DC motors through harmonic drive reduction units. Control is implemented on VME-bus based hardware, with a $300Hz$

servo rate and sample period variations of approximately ±5%. The robot is equipped with encoders having 360,000 encoder counts per revolution. The system configuration is displayed in Figure A.2.

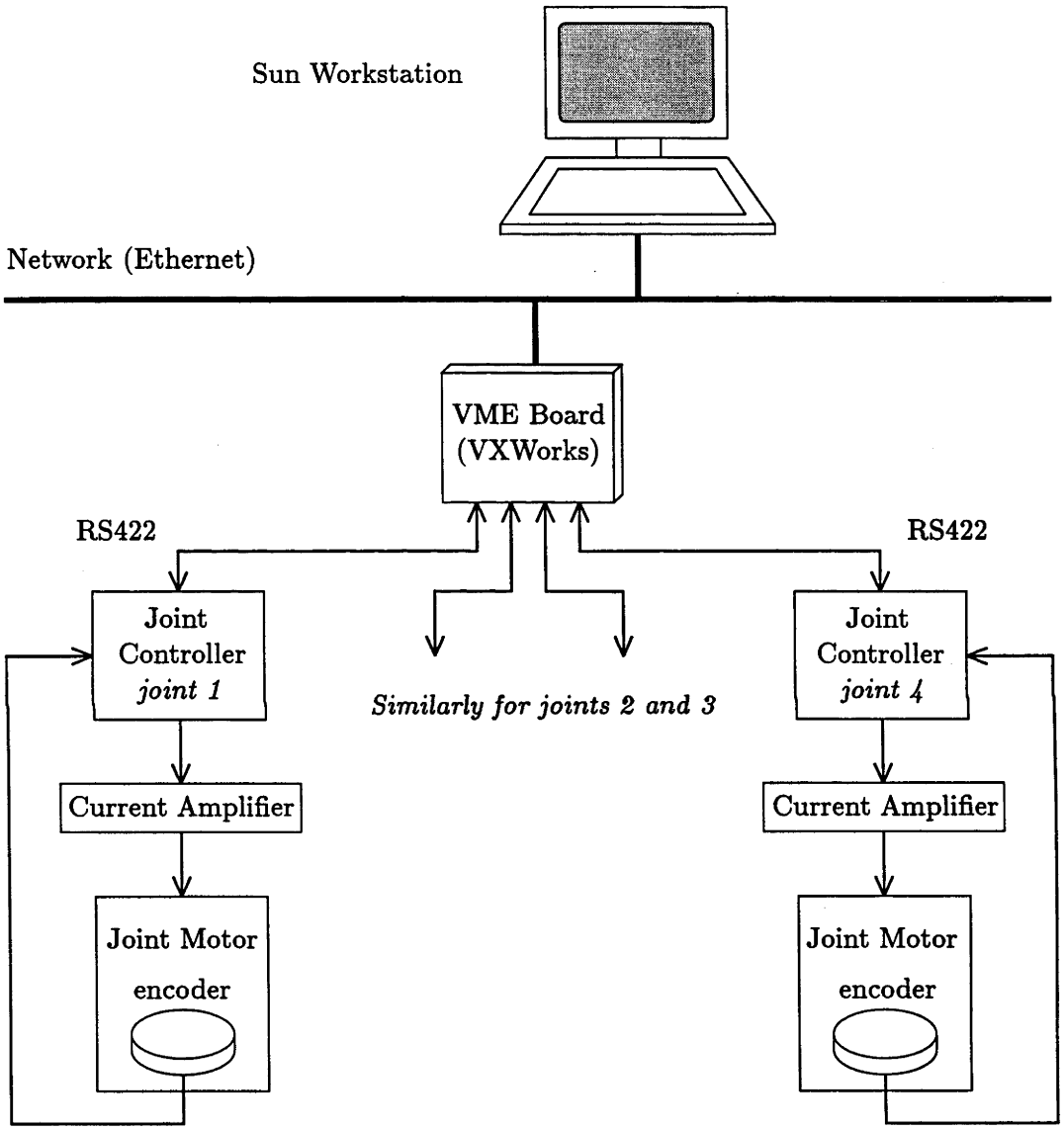


Figure A.2: Experimental System Configuration

For simplicity, and in all our experimental work, we restrict the motion to be planar, driven by only the first and second joint motors.

The general model for the planar dynamics of the SCARA is

$$\hat{\mathbf{M}}(\mathbf{q})\ddot{\mathbf{q}} + \hat{\mathbf{v}}(\mathbf{q},\dot{\mathbf{q}}) + \hat{\mathbf{d}}(\dot{\mathbf{q}}) = \boldsymbol{\tau}\,, \tag{A.1}$$

where the mass matrix, coriolis-centrifugal and friction vectors are

$$\hat{\mathbf{M}}(\mathbf{q}) = \begin{bmatrix} h_{11} + g_{11}\cos(q_2) + f_{11}\sin(q_2) & h_{12} + g_{12}\cos(q_2) + f_{12}\sin(q_2) \\ h_{21} + g_{21}\cos(q_2) + f_{21}\sin(q_2) & h_{22} \end{bmatrix}, \tag{A.2}$$

$$\hat{\mathbf{v}}(\mathbf{q},\dot{\mathbf{q}}) = \begin{bmatrix} v_{1s}\dot{q}_1\dot{q}_2\sin(q_2) + w_{1s}\dot{q}_2^2\sin(q_2) + v_{1c}\dot{q}_1\dot{q}_2\cos(q_2) + w_{1c}\dot{q}_2^2\cos(q_2) \\ v_{2s}\dot{q}_1^2\sin(q_2) + w_{2c}\dot{q}_1^2\cos(q_2) \end{bmatrix} \tag{A.3}$$

and

$$\hat{\mathbf{d}}(\dot{\mathbf{q}}) = \begin{bmatrix} f_{1p}\dot{q}_1^+ + b_{1p}\dot{q}_1\dot{q}_1^+ + f_{1n}\dot{q}_1^- + b_{1n}\dot{q}_1\dot{q}_1^- \\ f_{2p}\dot{q}_2^+ + b_{2p}\dot{q}_2\dot{q}_2^+ + f_{2n}\dot{q}_2^- + b_{2n}\dot{q}_2\dot{q}_2^- \end{bmatrix}, \tag{A.4}$$

respectively, and where

$$\dot{q}_i^+ = \begin{cases} 1 & \dot{q}_i > 0 \\ 0 & \dot{q}_i \le 0 \end{cases}, \quad \dot{q}_i^- = \begin{cases} 0 & \dot{q}_i > 0 \\ 1 & \dot{q}_i \le 0 \end{cases},$$

Note that the model includes all rigid body inertial effects, that the friction is modelled for each joint independently with coulomb and viscous terms, and that the model contains no gravity terms.

Software to identify the model parameters was developed by Paul Logothetis based on a method of least squares parameter identification [45].

The torque limits for this robot are due to current and voltage limits, and are given by

$$\left.\begin{aligned} \underline{\tau}(\dot{\mathbf{q}}) &= \max\left\{-Ki_{lim}, -\frac{Kv_{lim}}{R} - \frac{K^2}{R}\dot{\mathbf{q}}\right\} \\ \overline{\tau}(\dot{\mathbf{q}}) &= \min\left\{+Ki_{lim}, +\frac{Kv_{lim}}{R} - \frac{K^2}{R}\dot{\mathbf{q}}\right\} \end{aligned}\right\}, \tag{A.5}$$

where i_{lim} and v_{lim} represent the limits on the current and voltage, and where $K = 51.57NmA^{-1}$ is the motor constant and $R = 2.7\Omega$ is the resistance of the motors.

Bibliography

[1] C. Abdallah, D. Dawson, P. Dorato, and M. Jamshidi. Survey of robust control for rigid robots. In *IEEE Control Systems*, volume 11 (2), pages 24–30, February 1991.

[2] H. Arai and K. Tanie. Real-time path tracking with torque limits by using a disturbance observer. In *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, pages 1859–1865, San Diego, California, USA, May 1994.

[3] H. Arai, K. Tanie, and S Tachi. Path tracking control of a manipulator considering torque saturation. In *IEEE Transactions on Industrial Electronics*, volume 41 (1), pages 25–31, February 1994.

[4] A. Astolfi and L. Guzzella. Robust control of nonlinear systems - an $\mathcal{H}_\infty$ approach. *Pre-print*, January 1988.

[5] M. Bardi and M. Falcone. An approximation scheme for the minimum-time function. In *SIAM Journal of Control and Optimization*, volume 28 (4), pages 950–965, July 1990.

[6] P.R. Bélanger. Estimation of angular velocity and acceleration from shaft encoder measurements. In *Proceedings of the 1992 IEEE International Conference on Robotics and Automation*, pages 585–592, Nice, France, May 1992.

[7] R. Bellman. *Dynamic Programming.* Princeton University Press, Princeton, NJ, 1957.

[8] J.E. Bobrow, S. Dubowsky, and J.S. Gibson. Time-optimal control of robotic manipulators along specified paths. In *The International Journal of Robotics Research*, volume 4 (3), pages 3–17, Fall 1985.

[9] J. Butler and M. Tomizuka. A suboptimal reference generation technique for robotic manipulators following specified paths. In *Transactions of the American Society of Mechanical Engineers*, volume 114, pages 524–527, September 1992.

[10] A.J. Cahill, M.R. James, J.C. Kieffer, and D. Williamson. Remarks on the application of dynamic programming to the optimal path timing of robot manipulators. *Submitted to International Journal of Nonlinear and Robust Control*, August 1995.

[11] A.J. Cahill, J.C. Kieffer, and M.R. James. Robust time-optimal path tracking: Theory and experiments. *Submitted to IEEE Transactions on Robotics and Automation*, December 1995.

[12] A.J. Cahill, M.R. James, J.C. Kieffer, and D. Williamson. Optimal path timing of robot manipulators via dynamic programming. In *International Workshop on Nonlinear Systems and Adaptive Control*, Sydney, Australia, September 1994.

[13] A.J. Cahill and J.C. Kieffer. Fast pick and place at robot singularities. In *Proceedings of the 1995 IEEE International Conference on Robotics and Automation*, pages 2236–2241, Nagoya, Japan, May 1995.

[14] A.J. Cahill, J.C. Kieffer, and M.R. James. Implementation issues in the time-optimal control of robot manipulators along specified paths. In *Proceedings of the 1995 Conference of the Australian Robot Association*, pages 390–402, Melbourne, Australia, July 1995.

[15] A.J. Cahill, J.C. Kieffer, and M.R. James. On representing robot modelling errors as disturbances in joint accelerations: Theory and experiment. *To appear 34th IEEE International Conference on Decision and Control*, New Orleans, USA, December 1995.

[16] A.J. Cahill, J.C. Kieffer, and M.R. James. Time-optimal path tracking to a specified tolerance in the presence of plant uncertainty. *To appear IASTED International Conference on Applications of Control and Robotics*, Orlando, Florida, USA, January 1996.

[17] A.J. Cahill, J.C. Kieffer, and M.R. James. Robust time-optimal trajectory planning for robot manipulators. *Submitted to 1996 IEEE International Conference on Robotics and Automation*, Minneapolis, USA, September 1995.

[18] A.J. Cahill, J.C. Kieffer, and M.R. James. Closed-loop trajectory generation for robust time-optimal path tracking. *Submitted to 1996 IEEE International Conference on Robotics and Automation*, Minneapolis, USA, September 1995.

[19] B-S. Chen, T-S. Lee, and J-H. Feng. A nonlinear $\mathcal{H}_\infty$ control design in robotic systems under parametric perturbation and external disturbance. In *International Journal of Control*, volume 59 (2), pages 439–461, February 1994.

[20] Y. Chen, S.Y.-P. Chien, and A.A.Desrochers. General structure of time-optimal control of robotic manipulators moving along prescribed paths. In *International Journal of Control*, volume 56 (4), pages 767–782, October 1992.

[21] J.J. Craig. *Introduction to Robotics.* Addison Wesley, Reading, MA, 1989.

[22] O. Dahl. Path-constrained robot control. In *PhD Thesis.* Department of Automatic Control, Lund Institute of Technology, Lund, Sweden, 1992.

[23] O. Dahl. Path-constrained robot control with limited torques - experimental evaluation. In *IEEE Transactions on Robotics and Automation*, volume 10 (5), pages 658–669, October 1994.

[24] O. Dahl and L. Nielsen. Torque-limited path following by on-line trajectory time scaling. In *IEEE Transactions on Robotics and Automation*, volume 6 (5), pages 554–561, October 1990.

[25] M.W.M.G. Dissanayake. High-speed positioning of robot manipulators. In *Journal of Systems Engineering*, volume 1 (2), pages 103–113, 1991.

[26] M.W.M.G. Dissanayake, C.J. Goh, and N. Phan-Thien. Time-optimal trajectories for robot manipulators. In *Robotica*, volume 9 (2), pages 131–138, April 1991.

[27] I. Capuzzo Dolcetta and M. Falcone. Discrete dynamic programming and viscosity solutions of the bellman equation. *Pre-print*, January 1988.

[28] P. Dupuis and M.R. James. Rates of convergence for approximation schemes in optimal control. *Submitted to SIAM Journal of Control and Optimization*, 1994.

[29] L.C. Evans and M.R. James. The Hamilton-Jacobi-Bellman equation for time optimal control. In *SIAM Journal of Control and Optimization*, volume 27 (6), pages 1477–1489, November 1989.

[30] W.H. Fleming and R.W. Rishel. *Deterministic and Stochastic Optimal Control.* Springer Verlag, New York, NY, 1975.

[31] W.H. Fleming and H.M. Soner. *Controlled Markov Processes and Viscosity Solutions.* Springer Verlag, New York, NY, 1993.

[32] H.P. Geering, L. Guzzella, S.A.R. Hepner, and C.H. Odner. Time-optimal motions of robots in assembly tasks. In *IEEE Transactions on Automatic Control*, volume AC-31 (6), pages 512–518, June 1986.

[33] J.M. Hollerbach. Dynamic scaling of manipulator trajectories. In *Journal of Dynamic Systems, Measurement, and Control*, volume 106, pages 102–106, March 1984.

[34] J. Huang and Y. Chen. Robust time-optimal control of robotic manipulators under payload uncertainty. In *Proceedings of the 33rd IEEE International Conference on Decision and Control*, pages 2135–2136, Lake Buena Vista, Florida, USA, December 1994.

[35] K.H. Hunt. Special configurations of robot arms via screw theory (part 1). In *Robotica*, volume 4 (3), pages 171–179, July 1986.

[36] M. Kahn and B. Roth. The near minimum-time control of open-loop articulated kinematic chains. In *Journal of Dynamic Systems, Measurement and Control*, volume 93, pages 164–172, September 1971.

[37] J. Kieffer. Bifurcations and isolated singularities for robots and mechanisms. In *IEEE Transactions on Robotics and Automation*, volume 10 (1), pages 1–10, February 1994.

[38] J. Kieffer and J. Lenarcic. On the exploitation of mechanical advantage near robot singularities. In *Proceedings of the 3rd International Workshop on Advances in Robot Kinematics (3ARK)*, pages 65–72, Ferrara, Italy, 1992.

[39] B.K. Kim and K.G. Shin. Minimum-time path planning for robot arms and their dynamics. In *IEEE Transactions on Systems, Man, and Cybernetics*, volume SMC-15 (2), pages 213–223, April 1985.

[40] B.K. Kim and K.G. Shin. Suboptimal control of industrial manipulators with a weighted minimum time-fuel criterion. In *IEEE Transactions on Automatic Control*, volume AC-30 (1), pages 1–10, January 1985.

[41] H.J. Kushner and P.G. Dupuis. *Numerical Methods for Stochastic Control Problems in Continuous Time.* Springer Verlag, New York, NY, 1992.

[42] C.-S. Lin, P.R. Chang, and J.Y.S. Luh. Formulation and optimization of cubic polynomial trajectories for industrial robots. In *Proceedings of the 21st IEEE International Conference on Decision and Control*, pages 330–335, Orlando, Florida, USA, December 1982.

[43] C.-S. Lin, P.R. Chang, and J.Y.S. Luh. Formulation and optimization of cubic polynomial trajectories for industrial robots. In *IEEE Transactions on Automatic Control*, volume AC-28, pages 1066–1074, December 1983.

[44] G.-J. Liu and A.A. Goldenberg. Robust control of robot manipulators based on dynamics decomposition. In *Proceedings of the 33rd IEEE International Conference on Decision and Control*, pages 822–823, Lake Buena Vista, Florida, USA, December 1994.

[45] P. Logothetis and J.C. Kieffer. On the torque filtering scheme for eliminating acceleration measurement for robot identification. In *Proceedings of the 1995 Conference of the Australian Robot Association*, pages 100–107, Melbourne, Australia, July 1995.

[46] Z. Lu, K.B. Shimoga, and A.A Goldenberg. Experimental determination of dynamic parameters of robotic arms. In *Journal of Robotic Systems*, volume 10 (8), pages 1009–1029, December 1993.

[47] J.Y.S. Luh and C.-S. Lin. Optimum path planning for mechanical manipulators. In *Journal of Dynamic Systems, Measurement, and Control*, volume 102, pages 142–151, June 1981.

[48] J.Y.S. Luh, M.W. Walker, and R. Paul. Resolved-acceleration control of mechanical manipulators. In *IEEE Transactions on Automatic Control*, volume AC-25 (3), pages 468–474, June 1980.

[49] V. Lumelsky. *Robot Control - Dynamics, Motion Planning, and Analysis.* Motion planning. IEEE Press, 1993.

[50] S. Lyashevskiy and Y. Chen. Robust time-optimal feedback control of robotic manipulators with motor dynamics and parameter variations. In *Proceedings of the 33rd IEEE International Conference on Decision and Control*, pages 820–821, Lake Buena Vista, Florida, USA, December 1994.

[51] F. Pfieffer and R. Johanni. A concept for manipulator trajectory planning. In *IEEE Transactions on Robotics and Automation*, volume RA-3 (2), pages 115–123, April 1987.

[52] L.S. Pontryagin, V.G. Boltyanskii, R.V. Gamkriledze, and E.F. Mishenko. *The Mathematical Theory of Optimal Processes.* Interscience, New York, NY, 1962.

[53] C. Samson. Robust control of a class of nonlinear systems and applications to robotics. In *International Journal of Adaptive Control and Signal Processing*, volume 1 (1), pages 49–68, September 1987.

[54] Z. Shiller. On singular time-optimal control along specified paths. In *IEEE Transactions on Robotics and Automation*, volume 10 (4), pages 561–566, August 1994.

[55] Z. Shiller. Time-energy optimal control of articulated systems with geometric path constraints. In *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, pages 2680–2685, San Diego, California, USA, May 1994.

[56] Z. Shiller and S. Dubowsky. Robot path planning with obstacles, actuator, gripper and payload constraints. In *The International Journal of Robotics Research*, volume 8 (6), pages 3–18, December 1989.

[57] Z. Shiller and S. Dubowsky. On computing the global time-optimal motions of robotic manipulators in the presence of obstacles. In *IEEE Transactions on Robotics and Automation*, volume 7 (6), pages 785–797, December 1991.

[58] Z. Shiller and H-H. Lu. Computation of path constrained time optimal motions with dynamic singularities. In *Journal of Dynamic Systems, Measurement, and Control*, volume 114, pages 34–40, March 1992.

[59] K.G. Shin. *Robot Control - Dynamics, Motion Planning, and Analysis.* Time-optimal control of robots. IEEE Press, 1993.

[60] K.G. Shin and N.D. McKay. Minimum-time control of robotic manipulators with geometric path constraints. In *IEEE Transactions on Automatic Control*, volume AC-30 (6), pages 531–541, June 1985.

[61] K.G. Shin and N.D. McKay. A dynamic programming approach to trajectory planning of robotic manipulators. In *IEEE Transactions on Automatic Control*, volume AC-31 (6), pages 491–500, June 1986.

[62] K.G. Shin and N.D. McKay. Robust trajectory planning for robotic manipulators under payload uncertainties. In *IEEE Transactions on Automatic Control*, volume AC-32 (12), pages 1044–1054, December 1987.

[63] S. Singh and M.C. Leu. Optimal trajectory planning and control of robotic manipulators. In *Proceedings of the Japan-USA Symposium on Flexible Automation*, 1986.

[64] S. Singh and M.C. Leu. Optimal trajectory generation for robotic manipulators using dynamic programming. In *Journal of Dynamic Systems, Measurement, and Control*, volume 109, pages 88–96, June 1987.

[65] J-J.E. Slotine. The robust control of robot manipulators. In *The International Journal of Robotics Research*, volume 4 (2), pages 49–64, Summer 1985.

[66] J-J.E. Slotine and H.S. Yang. Improving the efficiency of time-optimal path-following algorithms. In *IEEE Transactions on Robotics and Automation*, volume 5 (1), pages 118–124, February 1989.

[67] E.D. Sontag and H.J. Sussman. Time-optimal control of manipulators. In *Proceedings of the 1986 IEEE International Conference on Robotics and Automation*, pages 1692–1697, San Francisco, USA, April 1986.

[68] H.Y. Tam. Minimum time closed-loop tracking of a specified path by robot. In *Proceedings of the 29th IEEE International Conference on Decision and Control*, pages 3132–3137, Honolulu, Hawaii, December 1990.

[69] M. Tarkiainen and Z. Shiller. Time-optimal motions of manipulators with actuator dynamics. In *Proceedings of the 1993 IEEE International Conference on Robotics and Automation*, pages 725–730, Atlanta, Georgia, USA, May 1993.

[70] A. Tesfaye and M. Tomizuka. Robust MIMO model following with application to trajectory motion control of robot arms. In *Proceedings of the 33rd IEEE International Conference on Decision and Control*, pages 814–819, Lake Buena Vista, Florida, USA, December 1994.

[71] J.T. Wen and A. Desrochers. Sub-time-optimal control strategies for robotic manipulators. In *Proceedings of the 1986 IEEE International Conference on Robotics and Automation*, pages 402–406, San Francisco, USA, April 1986.